IT Security – A Comprehensive Treatment

An NUS Seminar, May 2014

Dines Bjørner
Fredsvej 11, DK–2840 Hole, Denmark

March 22, 2014

1 Introduction

We analyse the domain of IT systems and 'add' to that domain the concept of IT Security Rules (and Regulations). The analysis is done, first informally, then formally. The informal analysis and its presentation follows the 'dogmas' set out in Vol.3 of Software Engineering. The formal presentation follows the principles and techniques and uses the tools outlined in Vols.1-2 of the afore-mentioned book.

We choose the following sequence of analysis and synthesis actions: First we bring excerpts from the ISO Standard: INTERNATIONAL ISO/IEC STANDARD 17799: Information technology: security techniques — code of practice for information security management. On the basis of these rather cursory excerpts but also on the basis of a more comprehensive analysis — both of which we do not show — we postulate in five sections (Sects. 5–6) a domain model for IT systems.

The formal model has two components: A formal model of system configurations: states and contexts; and a formal model of the "codes of practice for information security management". The former model is a conventional, software engineering model of "a system". Maybe there are some novel aspects that enable us to perform spatial (or diagrammatic) reasoning. Maybe existing work on spatial reasoning ought be consulted . The latter model is a rather conventional model of the semantics of well formed formulas (*wff*s) in logic — without including modal operations — curiously absent, it seems, from the "ISO Code of Practice". The assumption being made here is that all "implementation guideline" statements of the "ISO Code of Practice" can be expressed in some (first ?) order predicate calculus.

This approach to the modelling of a "code of practice for information security management" is tentative. That is, it is an experiment. Maybe we succeed. Maybe we do not. The work reported here is thus of the following nature: it is experimental, it aims at understanding the domain of IT systems and of the related "code of practice for information security management". and of testing our principles and techniques of domain engineering with this "testing" being carried out in Sects. 5–6. If we get a formal model of the ISO (standard) “code of practice for information security management” that reveals that can be used to question this “code of practice”, that can be used for “prediction”, and on the basis of which we can implement computing and communication) systems support for this “practice” then we would claim the experiment for being successful.

2 An Example Set of IT System Codes of Practice

We quote extensively from INTERNATIONAL ISO/IEC STANDARD 17799: Information technology: security techniques — code of practice for information security management.

2.1 [6] Organisation of information security

2.1.1 [6.1] Internal Organisation

[6.1.1] Management commitment to information security. Control:

Management should actively support security within the organization through clear direction, demonstrated commitment, explicit assignment, and acknowledgment of information security responsibilities.

Implementation guidance:

Management should:

1. *ensure that information security goals are identified, meet the organizational requirements, and are integrated in relevant processes;*
2. *formulate, review, and approve information security policy;*
3. *review the effectiveness of the implementation of the information security policy;*
4. *provide clear direction and visible management support for security initiatives;*
5. *provide the resources needed for information security;*
6. *approve assignment of specific roles and responsibilities for information security across the organization;*
7. *initiate plans and programs to maintain information security awareness;*
8. *ensure that the implementation of information security controls is co-ordinated across the organization (see 6.1.2).*

[6.1.2] Information security co-ordination. Control:

Information security activities should be co-ordinated by representatives from different parts of the organization with relevant roles and job functions.

Implementation guidance:

Typically, information security co-ordination should involve the co-operation and collaboration of managers, users, administrators, application designers, auditors and security personnel, and specialist skills in areas such as insurance, legal issues, human resources, IT and risk management.

This activity should:

1. *ensure that security activities are executed in compliance with the information security policy;*

2. *identify how to handle non-compliances;*

3. *approve methodologies and processes for information security, e.g. risk assessment, information classification;*

4. *identify significant threat changes and exposure of information and information processing facilities to threats;*

5. *assess the adequacy and co-ordinate the implementation of information security controls;*

6. *effectively promote information security education, training and awareness throughout the organization;*

7. *evaluate information received from the monitoring and reviewing of information security incidents, and recommend appropriate actions in response to identified information security incidents.*

2.1.2 [6.2] External parties

Objective: *(1) To maintain the security of the organization's information and information processing facilities that are accessed, processed, communicated to, or managed by external parties. (2) The security of the organization's information and information processing facilities should not be reduced by the introduction of external party products or services. (3) Any access to the organization's information processing facilities and processing and communication of information by external parties should be controlled. (4) Where there is a business need for working with external parties that may require access to the organization's information and information processing facilities, or in obtaining or providing a product and service from or to an external party, a risk assessment should be carried out to determine security implications and control requirements. Controls should be agreed and defined in an agreement with the external party.*

[6.2.1] Identification of risks related to external parties. Control:

The risks to the organization's information and information processing facilities from business processes involving external parties should be identified and appropriate controls implemented before granting access.

Implementation guidance:

Where there is a need to allow an external party access to the information processing facilities or information of an organization, a risk assessment (see also Section 4) should be carried out to identify any requirements for specific controls. The identification of risks related to external party access should take into account the following issues:

1. *the information processing facilities an external party is required to access;*
2. *the type of access the external party will have to the information and information processing facilities, e.g.:*
   (a) *physical access, e.g. to offices, computer rooms, filing cabinets;*
   (b) *logical access, e.g. to an organization's databases, information systems;*
   (c) *network connectivity between the organization's and the external partys network(s), e.g. permanent connection, remote access;*
   (d) *whether the access is taking place on-site or off-site;*
3. *the value and sensitivity of the information involved, and its criticality for business operations;*
4. *the controls necessary to protect information that is not intended to be accessible by external parties;*
5. *the external party personnel involved in handling the organization's information;*
6. *how the organization or personnel authorized to have access can be identified, the authorization verified, and how often this needs to be reconfirmed;*
7. *the different means and controls employed by the external party when storing, processing, communicating, sharing and exchanging information;*
8. *the impact of access not being available to the external party when required, and the external party entering or receiving inaccurate or misleading information;*
9. *practices and procedures to deal with information security incidents and potential damages, and the terms and conditions for the continuation of external party access in the case of an information security incident;*
10. *legal and regulatory requirements and other contractual obligations relevant to the external party that should be taken into account;*

11. *how the interests of any other stakeholders may be affected by the arrangements.*

Access by external parties to the organization's information should not be provided until the appropriate controls have been implemented and, where feasible, a contract has been signed defining the terms and conditions for the connection or access and the working arrangement. Generally, all security requirements resulting from work with external parties or internal controls should be reflected by the agreement with the external party (see also 6.2.2 and 6.2.3).

It should be ensured that the external party is aware of their obligations, and accepts the responsibilities and liabilities involved in accessing, processing, communicating, or managing the organization's information and information processing facilities.

Other information:

Information might be put at risk by external parties with inadequate security management. Controls should be identified and applied to administer external party access to information processing facilities. For example, if there is a special need for confidentiality of the information, non-disclosure agreements might be used.

Organizations may face risks associated with inter-organizational processes, management, and communication if a high degree of outsourcing is applied, or where there are several external parties involved.

The controls 6.2.2 and 6.2.3 cover different external party arrangements, e.g. including:

1. *service providers, such as ISPs, network providers, telephone services, maintenance and support services;*
2. *managed security services;*
3. *customers;*
4. *outsourcing of facilities and/or operations, e.g. IT systems, data collection services, call centre operations;*
5. *management and business consultants, and auditors;*
6. *developers and suppliers, e.g. of software products and IT systems;*
7. *cleaning, catering, and other outsourced support services;*
8. *temporary personnel, student placement, and other casual short-term appointments.*

Such agreements can help to reduce the risks associated with external parties.

2.2 [7] Asset management

2.2.1 [7.1] Responsibility for assets

[7.1.1] Inventory of assets. **Control:** *All assets should be clearly identified and an inventory of all important assets drawn up and maintained.*

Implementation guidance:

An organization should identify all assets and document the importance of these assets. The asset inventory should include all information necessary in order to recover from a disaster, including type of asset, format, location, backup information, license information, and a business value. The inventory should not duplicate other inventories unnecessarily, but it should be ensured that the content is aligned.

In addition, ownership (see 7.1.2) and information classification (see 7.2) should be agreed and documented for each of the assets. Based on the importance of the asset, its business value and its security classification, levels of protection commensurate with the importance of the assets should be identified (more information on how to value assets to represent their importance can be found in ISO/IEC TR 13335-3).

Other information: *There are many types of assets, including:*

1. *information: databases and data files, contracts and agreements, system documentation, research information, user manuals, training material, operational or support procedures, business continuity plans, fallback arrangements, audit trails, and archived information;*
2. *software assets: application software, system software, development tools, and utilities;*
3. *physical assets: computer equipment, communications equipment, removable media, and other equipment;*
4. *services: computing and communications services, general utilities, e.g. heating, lighting, power, and air-conditioning;*
5. *people, and their qualifications, skills, and experience;*
6. *intangibles, such as reputation and image of the organization.*

Inventories of assets help to ensure that effective asset protection takes place, and may also be required for other business purposes, such as health and safety, insurance or financial (asset management) reasons. The process of compiling an inventory of assets is an important prerequisite of risk management (see also Section 4).

2.3 [8] Human resources security

2.3.1 [8.1] Prior to employment

(Explanation: The word 'employment' is meant here to cover all of the following different situations: employment of people (temporary or longer lasting), appointment of job roles, changing of job roles, assignment of contracts, and the termination of any of these arrangements.)

Objective: *To ensure that employees, contractors and third party users understand their responsibilities, and are suitable for the roles they are considered for, and to reduce the risk of theft, fraud or misuse of facilities.*

Security responsibilities should be addressed prior to employment in adequate job descriptions and in terms and conditions of employment.

All candidates for employment, contractors and third party users should be adequately screened, especially for sensitive jobs.

Employees, contractors and third party users of information processing facilities should sign an agreement on their security roles and responsibilities.

[8.1.1] Roles and responsibilities. **Control:**

Security roles and responsibilities of employees, contractors and third party users should be defined and documented in accordance with the organization's information security policy.

Implementation guidance:

Security roles and responsibilities should include the requirement to:

1. *implement and act in accordance with the organizations information security policies (see 5.1);*
2. *protect assets from unauthorized access, disclosure, modification, destruction or interference;*
3. *execute particular security processes or activities;*
4. *ensure responsibility is assigned to the individual for actions taken;*
5. *report security events or potential events or other security risks to the organization.*

Security roles and responsibilities should be defined and clearly communicated to job candidates during the pre-employment process.

2.4 [9] Physical and environmental security

2.4.1 [9.1] Secure areas

Objective: *To prevent unauthorized physical access, damage, and interference to the organization's premises and information. Critical or sensitive information processing facilities should be housed in secure areas, protected by defined security perimeters, with appropriate security barriers and entry controls. They should be physically protected from unauthorized access, damage, and interference. The protection provided should be commensurate with the identified risks.*

[9.1.1] Physical security perimeter. **Control:** *Security perimeters (barriers such as walls, card controlled entry gates or manned reception desks) should be used to protect areas that contain information and information processing facilities.*

Implementation guidance:

The following guidelines should be considered and implemented where appropriate for physical security perimeters:

1. *security perimeters should be clearly defined, and the siting and strength of each of the perimeters should depend on the security requirements of the assets within the perimeter and the results of a risk assessment;*

2. *perimeters of a building or site containing information processing facilities should be physically sound (i.e. there should be no gaps in the perimeter or areas where a break-in could easily occur); the external walls of the site should be of solid construction and all external doors should be suitably protected against unauthorized access with control mechanisms, e.g. bars, alarms, locks etc; doors and windows should be locked when unattended and external protection should be considered for windows, particularly at ground level;*

3. *a manned reception area or other means to control physical access to the site or building should be in place; access to sites and buildings should be restricted to authorized personnel only;*

4. *physical barriers should, where applicable, be built to prevent unauthorized physical access and environmental contamination;*

5. *all fire doors on a security perimeter should be alarmed, monitored, and tested in conjunction with the walls to establish the required level of resistance in accordance to suitable regional, national, and international standards; they should operate in accordance with local fire code in a failsafe manner;*

6. *suitable intruder detection systems should be installed to national, regional or international standards and regularly tested to cover all external doors and accessible windows; unoccupied areas should be alarmed at all times; cover should also be provided for other areas, e.g. computer room or communications rooms;*

7. *information processing facilities managed by the organization should be physically separated from those managed by third parties.*

[9.1.2] Physical entry controls. **Control:** *Secure areas should be protected by appropriate entry controls to ensure that only authorized personnel are allowed access.*

Implementation guidance:

1. *the date and time of entry and departure of visitors should be recorded, and all visitors should be supervised unless their access has been previously approved; they should only be granted access for specific, authorized purposes and should be issued with instructions on the security requirements of the area and on emergency procedures.*

2. *access to areas where sensitive information is processed or stored should be controlled and restricted to authorized persons only; authentication controls, e.g. access control card plus PIN, should be used to authorize and validate all access; an audit trail of all access should be securely maintained;*

3. *all employees, contractors and third party users and all visitors should be required to wear some form of visible identification and should immediately notify security personnel if they encounter un-escorted visitors and anyone not wearing visible identification;*

4. *third party support service personnel should be granted restricted access to secure areas or sensitive information processing facilities only when required; this access should be authorized and monitored;*

5. *access rights to secure areas should be regularly reviewed and updated, and revoked when necessary (see 8.3.3).*

[9.1.3] **Securing offices, rooms, and facilities.** **Control** *Physical security for offices, rooms, and facilities should be designed and applied.*

Implementation guidance: *The following guidelines should be considered to secure offices, rooms, and facilities:*

1. *account should be taken of relevant health and safety regulations and standards;*

2. *key facilities should be sited to avoid access by the public;*

3. *where applicable, buildings should be unobtrusive and give minimum indication of their purpose, with no obvious signs, outside or inside the building identifying the presence of information processing activities;*

4. *directories and internal telephone books identifying locations of sensitive information processing facilities should not be readily accessible by the public.*

[9.1.4] **Protecting against external and environmental threats.** **Control:** *Physical protection against damage from fire, flood, earthquake, explosion, civil unrest, and other forms of natural or man-made disaster should be designed and applied.*

Implementation guidance:

Consideration should be given to any security threats presented by neighboring premises, e.g. a fire in a neighbouring building, water leaking from the roof or in floors below ground level or an explosion in the street.

1. *hazardous or combustible materials should be stored at a safe distance from a secure area. Bulk supplies such as stationery should not be stored within a secure area;*

2. *fallback equipment and back-up media should be sited at a safe distance to avoid damage from a disaster affecting the main site;*

3. *appropriate fire fighting equipment should be provided and suitably placed.*

[9.1.5] Working in secure areas. **Control:** *Physical protection and guidelines for working in secure areas should be designed and applied.*

Implementation guidance:

1. *personnel should only be aware of the existence of, or activities within, a secure area on a need to know basis;*
2. *unsupervised working in secure areas should be avoided both for safety reasons and to prevent opportunities for malicious activities;*
3. *vacant secure areas should be physically locked and periodically checked;*
4. *photographic, video, audio or other recording equipment, such as cameras in mobile devices, should not be allowed, unless authorized;*

The arrangements for working in secure areas include controls for the employees, contractors and third party users working in the secure area, as well as other third party activities taking place there.

[9.1.6] Public access, delivery, and loading areas. **Control:** *Access points such as delivery and loading areas and other points where unauthorized persons may enter the premises should be controlled and, if possible, isolated from information processing facilities to avoid unauthorized access.*

2.4.2 [9.2] Equipment security

Objective: *To prevent loss, damage, theft or compromise of assets and interruption to the organization's activities. Equipment should be protected from physical and environmental threats. Protection of equipment (including that used off-site, and the removal of property) is necessary to reduce the risk of unauthorized access to information and to protect against loss or damage. This should also consider equipment siting and disposal. Special controls may be required to protect against physical threats, and to safeguard supporting facilities, such as the electrical supply and cabling infrastructure.*

[9.2.1] Equipment siting and protection. **Control:** *Equipment should be sited or protected to reduce the risks from environmental threats and hazards, and opportunities for unauthorized access.*

[9.2.2] Supporting utilities. **Control:** *Equipment should be protected from power failures and other disruptions caused by failures in supporting utilities.*

[9.2.3] Cabling security. **Control:** *Power and telecommunications cabling carrying data or supporting information services should be protected from interception or damage.*

[9.2.4] **Equipment maintenance.** **Control:** *Equipment should be correctly maintained to ensure its continued availability and integrity.*

[9.2.5] **Security of equipment off-premises.** **Control:** *Security should be applied to off-site equipment taking into account the different risks of working outside the organization's premises.*

Implementation guidance: *Regardless of ownership, the use of any information processing equipment outside the organization's premises s must be authorized by management.*

1. *equipment and media taken off the premises should not be left unattended in public places; portable computers should be carried as hand luggage and disguised where possible when travelling;*
2. *manufacturers' instructions for protecting equipment should be observed at all times, e.g. protection against exposure to strong electromagnetic fields;*
3. *home-working controls should be determined by a risk assessment and suitable controls applied as appropriate, e.g. lockable filing cabinets, clear desk policy, access controls for computers and secure communication with the office (see also ISO/IEC 18028 Network Security);*
4. *adequate insurance cover should be in place to protect equipment off-site.*

Security risks, e.g. of damage, theft or eavesdropping, may vary considerably between locations and should be taken into account in determining the most appropriate controls.

2.5 [10] Communications and operations management

2.5.1 [10.1] Operational procedures and responsibilities

Objective: *To ensure the correct and secure operation of information processing facilities. Responsibilities and procedures for the management and operation of all information processing facilities should be established. This includes the development of appropriate operating procedures. Segregation of duties should be implemented, where appropriate, to reduce the risk of negligent or deliberate system misuse.*

[10.1.1] **Documented operating procedures.** **Control:** *Operating procedures should be documented, maintained, and made available to all users who need them.*

[10.1.2] **Change management.** **Control:** *Changes to information processing facilities and systems should be controlled.*

[10.1.4] Separation of development, test, and operational facilities. **Control:** *Development, test, and operational facilities should be separated to reduce the risks of unauthorised access or changes to the operational system.*

2.5.2 [10.4] Protection against malicious and mobile code

Objective: *To protect the integrity of software and information. Precautions are required to prevent and detect the introduction of malicious code and unauthorized mobile code. Software and information processing facilities are vulnerable to the introduction of malicious code, such as computer viruses, network worms, Trojan horses, and logic bombs. Users should be made aware of the dangers of malicious code. Managers should, where appropriate, introduce controls to prevent, detect, and remove malicious code and control mobile code.*

[10.4.1] Controls against malicious code. **Control:** *Detection, prevention, and recovery controls to protect against malicious code and appropriate user awareness procedures should be implemented.*

2.5.3 [10.5] Back-up

Objective:*To maintain the integrity and availability of information and information processing facilities. Routine procedures should be established to implement the agreed back-up policy and strategy for taking back-up copies of data and rehearsing their timely restoration.*

[10.5.1] Information back-up. **Control:** *Back-up copies of information and software should be taken and tested regularly in accordance with the agreed backup policy.*

2.5.4 [10.6] Network security management

Objective: *To ensure the protection of information in networks and the protection of the supporting infrastructure.*

The secure management of networks, which may span organizational boundaries, requires careful consideration to data-flow, legal implications, monitoring, and protection. Additional controls may also be required to protect sensitive information passing over public networks.

[10.6.1] Network controls. **Control:** *Networks should be adequately managed and controlled, in order to be protected from threats, and to maintain security for the systems and applications using the network, including information in transit.*

2.5.5 [10.7] Media handling

Objective: *To prevent unauthorized disclosure, modification, removal or destruction of assets, and interruption to business activities.*

Media should be controlled and physically protected.

Appropriate operating procedures should be established to protect documents, computer media (e.g. tapes, disks), input/output data and system documentation from unauthorized disclosure, modification, removal, and destruction.

[10.7.1] Management of removable media. **Control:** *There should be procedures in place for the management of removable media.*

[10.7.2] Disposal of media. **Control:** *Media should be disposed-of securely and safely when no longer required, using formal procedures.*

[10.7.3] Information handling procedures . **Control:** *Procedures for the handling and storage of information should be established to protect this information from unauthorized disclosure or misuse.*

[10.7.4] Security of system documentation. **Control:** *System documentation should be protected against unauthorized access.*

2.5.6 [10.8] Exchange of information

Objective: *To maintain the security of information and software exchanged within an organization and with any external entity.*

Exchanges of information and software between organizations should be based on a formal exchange policy, carried out in line with exchange agreements, and should be compliant with any relevant legislation (see clause 15).

Procedures and standards should be established to protect information and physical media containing information in transit.

[10.8.1] Information exchange policies and procedures. **Control:** *Formal exchange policies, procedures, and controls should be in place to protect the exchange of information through the use of all types of communication facilities.*

[10.8.3] Physical media in transit. **Control:** *Media containing information should be protected against unauthorized access, misuse or corruption during transportation beyond an organization's physical boundaries.*

Implementation guidance:

1. *reliable transport or couriers should be used;*
2. *a list of authorized couriers should be agreed with management;*
3. *procedures to check the identification of couriers should be developed;*

4. *packaging should be sufficient to protect the contents from any physical damage likely to arise during transit and in accordance with any manufacturers' specifications (e.g. for software), for example protecting against any environmental factors that may reduce the media's restoration effectiveness such as exposure to heat, moisture or electromagnetic fields;*

5. *controls should be adopted, where necessary, to protect sensitive information from unauthorized disclosure or modification; examples include:*

   (a) *use of locked containers;*

   (b) *delivery by hand;*

   (c) *tamper-evident packaging (which reveals any attempt to gain access);*

   (d) *in exceptional cases, splitting of the consignment into more than one delivery and dispatch by different routes.*

[10.8.4] Electronic messaging. **Control:** *Information involved in electronic messaging should be appropriately protected.*

Implementation guidance:

1. *protecting messages from unauthorized access, modification or denial of service;*

2. *ensuring correct addressing and transportation of the message;*

3. *general reliability and availability of the service;*

4. *legal considerations, for example requirements for electronic signatures;*

5. *obtaining approval prior to using external public services such as instant messaging or file sharing;*

6. *stronger levels of authentication controlling access from publicly accessible networks.*

2.5.7 [10.10] Monitoring

Objective: *To detect unauthorized information processing activities.*

Systems should be monitored and information security events should be recorded. Operator logs and fault logging should be used to ensure information system problems are identified.

An organization should comply with all relevant legal requirements applicable to its monitoring and logging activities.

System monitoring should be used to check the effectiveness of controls adopted and to verify conformity to an access policy model.

[10.10.1] Audit logging. **Control:** *Audit logs recording user activities, exceptions, and information security events should be produced and kept for an agreed period to assist in future investigations and access control monitoring.*

[10.10.2] Monitoring system use. **Control:** *Procedures for monitoring use of information processing facilities should be established and the results of the monitoring activities reviewed regularly.*

Implementation guidance: *The level of monitoring required for individual facilities should be determined by a risk assessment. An organisation should comply with all relevant legal requirements applicable to its monitoring activities.*

Areas that should be considered include:

1. *authorized access, including detail such as:*
   - *(a) the user ID;*
   - *(b) the date and time of key events;*
   - *(c) the types of events;*
   - *(d) the files accessed;*
   - *(e) the program/utilities used;*
2. *all privileged operations, such as:*
   - *(a) use of privileged accounts, e.g. supervisor, root, administrator;*
   - *(b) system start-up and stop;*
   - *(c) I/O device attachment/detachment;*
3. *unauthorized access attempts, such as:*
   - *(a) failed or rejected user actions;*
   - *(b) failed or rejected actions involving data and other resources;*
   - *(c) access policy violations and notifications for network gateways and firewalls;*
   - *(d) alerts from proprietary intrusion detection systems;*
4. *system alerts or failures such as:*
   - *(a) console alerts or messages;*
   - *(b) system log exceptions;*
   - *(c) network management alarms;*
   - *(d) alarms raised by the access control system;*
5. *changes to, or attempts to change, system security settings and controls.*

How often the results of monitoring activities are reviewed should depend on the risks involved. Risk factors that should be considered include the:

1. *criticality of the application processes;*
2. *value, sensitivity, and criticality of the information involved;*
3. *past experience of system infiltration and misuse, and the frequency of vulnerabilities being exploited;*
4. *extent of system interconnection (particularly public networks);*
5. *logging facility being de-activated.*

2.6 [11] Access control

2.6.1 [11.1] Business requirement for access control

Objective: *To control access to information. Access to information, information processing facilities, and business processes should be controlled on the basis of business and security requirements. Access control rules should take account of policies for information dissemination and authorization.*

[11.1.1] Access control policy. **Control:** *An access control policy should be established, documented, and reviewed based on business and security requirements for access.*

2.6.2 [11.2] User access management

Objective: *To ensure authorized user access and to prevent unauthorized access to information systems. Formal procedures should be in place to control the allocation of access rights to information systems and services.*

The procedures should cover all stages in the life-cycle of user access, from the initial registration of new users to the final de-registration of users who no longer require access to information systems and services. Special attention should be given, where appropriate, to the need to control the allocation of privileged access rights, which allow users to override system controls.

[11.2.1] User registration. **Control:** *There should be a formal user registration and de-registration procedure in place for granting and revoking access to all information systems and services.*

Implementation guidance:

1. *using unique user IDs to enable users to be linked to and held responsible for their actions; the use of group IDs should only be permitted where they are necessary for business or operational reasons, and should be approved and documented;*

2. *checking that the user has authorization from the system owner for the use of the information system or service; separate approval for access rights from management may also be appropriate;*
3. *checking that the level of access granted is appropriate to the business purpose (see 11.1) and is consistent with organizational security policy, e.g. it does not compromise segregation of duties (see 10.1.3);*
4. *giving users a written statement of their access rights;*
5. *requiring users to sign statements indicating that they understand the conditions of access;*
6. *ensuring service providers do not provide access until authorization procedures have been completed;*
7. *maintaining a formal record of all persons registered to use the service;*
8. *immediately removing or blocking access rights of users who have changed roles or jobs or left the organization;*
9. *periodically checking for, and removing or blocking, redundant user IDs and accounts (see 11.2.4);*
10. *ensuring that redundant user IDs are not issued to other users.*

Other information: *Consideration should be given to establish user access roles based on business requirements that summarize a number of access rights into typical user access profiles. Access requests and reviews (see 11.2.4) are easier managed at the level of such roles than at the level of particular rights.*

Consideration should be given to including clauses in personnel contracts and service contracts that specify sanctions if unauthorized access is attempted by personnel or service agents (see also 6.1.5, 8.1.3 and 8.2.3).

[11.2.2] Privilege management. **Control:** *The allocation and use of privileges should be restricted and controlled.*

[11.2.3] User password management. **Control:** *The allocation of passwords should be controlled through a formal management process.*

[11.2.4] Review of user access rights. **Control:** *Management should review users' access rights at regular intervals using a formal process.*

2.6.3 [11.4] Network access control

Objective: *To prevent unauthorized access to networked services. Access to both internal and external networked services should be controlled. User access to networks and network services should not compromise the security of the network services by ensuring:*

1. *appropriate interfaces are in place between the organization's network and networks owned by other organizations, and public networks;*
2. *appropriate authentication mechanisms are applied for users and equipment;*
3. *control of user access to information services in enforced.*

[11.4.1] Policy on use of network services. **Control:** *Users should only be provided with access to the services that they have been specifically authorized to use.*

[11.4.2] User authentication for external connections. **Control:** *Appropriate authentication methods should be used to control access by remote users.*

[11.4.3] Equipment identification in networks. **Control:** *Automatic equipment identification should be considered as a means to authenticate connections from specific locations and equipment.*

[11.4.4] Remote diagnostic and configuration port protection. **Control:** *Physical and logical access to diagnostic and configuration ports should be controlled.*

[11.4.5] Segregation in networks. **Control:** *Groups of information services, users, and information systems should be segregated on networks.*

2.6.4 [11.5] Operating system access control

Objective: *To prevent unauthorized access to operating systems. Security facilities should be used to restrict access to operating systems to authorized users.*

1. *authenticating authorized users, in accordance with a defined access control policy;*
2. *recording successful and failed system authentication attempts;*
3. *recording the use of special system privileges;*
4. *issuing alarms when system security policies are breached;*
5. *providing appropriate means for authentication;*
6. *where appropriate, restricting the connection time of users.*

[11.5.1] Secure log-on procedures. **Control:** *Access to operating systems should be controlled by a secure log-on procedure.*

2.7 [13] Information security incident management

2.7.1 [13.1] Reporting information security events and weaknesses

Objective: *To ensure information security events and weaknesses associated with information systems are communicated in a manner allowing timely corrective action to be taken.*

Formal event reporting and escalation procedures should be in place. All employees, contractors and third party users should be made aware of the procedures for reporting the different types of event and weakness that might have an impact on the security of organizational assets. They should be required to report any information security events and weaknesses as quickly as possible to the designated point of contact.

[13.1.1] Reporting information security events. **Control:** *Information security events should be reported through appropriate management channels as quickly as possible.*

2.7.2 [13.2] Management of information security incidents and improvements

Objective: *To ensure a consistent and effective approach is applied to the management of information security incidents.*

Responsibilities and procedures should be in place to handle information security events and weaknesses effectively once they have been reported. A process of continual improvement should be applied to the response to, monitoring, evaluating, and overall management of information security incidents.

Where evidence is required, it should be collected to ensure compliance with legal requirements.

[13.1.1] Reporting security weaknesses. **Control:**

All employees, contractors and third party users of information systems and services should be required to note and report any observed or suspected security weaknesses in systems or services.

3 The ISO Standard ISO/IEC 17799 Table-of-Contents

4 An Analysis of the ISO/IEC 17799 Code of Practice

A subset of the above subset:

[6.1.1] Management commitment to information security: .

- **The 'Code of Practice' Statement:**

  Management should:

  1. ensure that information security goals are identified, meet the organizational requirements, and are integrated in relevant processes;
  2. formulate, review, and approve information security policy;
  3. review the effectiveness of the implementation of the information security policy;
  4. provide clear direction and visible management support for security initiatives;
  5. provide the resources needed for information security;
  6. approve assignment of specific roles and responsibilities for information security across the organization;

7. initiate plans and programs to maintain information security awareness;
8. ensure that the implementation of information security controls is co-ordinated across the organization (see 6.1.2).

- **A Predicate Term Interpretation:**

    1. exists('information_security_goals')(system)

       ∧ exists('organizational_requirements')(system)

       ∧ does_meet(system('information_security_goals'),

       system('organizational_requirements'))

       ∧ is_integrated(system('information_security_goals'),

       system('system_processes'))

    2. exists('information_security_policy')(system)

       ∧ is_reviewed(system('information_security_policy'))

       ∧ is_approved(system('information_security_policy'))

    3. is_effective(system('information_security_policy'))

    4. exists('security_initiatives')(system)

       ∧ exists('directives')(system)

       ∧ is_visible((system('security_initiatives'))('management_support'))

    5. is_adequate(system('resources')),

       (resources(system('information_security_policy')))

    6. exists('role_assignment')(system('information_security'))

       ∧ exists('responsibilities')(system('information_security'))

    7. is_aware('information_security')(system)

       ⊃ exists('plans')(system('information_security'))

       ∧ exists('programs')(system('information_security'))

    8. exists('information_security_controls')(system)

       ⊃ is_coordinated('information_security_controls')(system)

- **Some Comments:**

    1. **The formal expression:**

       exists('information_security_goals')(system)

       ∧ exists('organizational_requirements')(system)

       ∧ does_meet(system('information_security_goals'),

       system('organizational_requirements'))

       ∧ is_integrated(system('information_security_goals'),

       system('system_processes'))

       Comments:

  - exists *names a rather general predicate.*
  - *It applies to a name n and the "entire"* system.
  - *It is thus assumed that this entire* system *will posses a* document *named n.*
  - *Thus* system(n) *"selects" that* document.
  - does_meet *names a predicate.*
  - *It applies to two documents.*
  - system('system_processes') *"selects" the current* system processes *— or, possibly, the possibly infinite set of all potential* system processes.
  - is_integrated *names a predicate.*
  - is_integrated *applies to a* document *and the (...)* system processes *and checks (somehow) that the entities designated by the* document *are* integrated *in these* processes.
  - *Note that the first argument of* is_integrated *is a* document *whereas the second argument is a* dynamic system entity.

2. **The formal expression:**

   exists('information_security_policy')(system)
   ∧ is_reviewed(system('information_security_policy'))
   ∧ is_approved(system('information_security_policy'))

   Comments:

   - *The assumption here is that the* document

     system('information_security_policy')

     possess at least the attributes of having been 'reviewed' and having been 'approved'.
   - *This entails two other assumptions: that that* document *is subject to the two corresponding $functions$*
     - review and
     - approve.

3. **The formal expression:**

   is_effective(system('information_security_policy'))

   Comments:

   - is_effective *names a predicate.*
   - *It applies to a document*
   - *and somehow determines whether it is* effective.

4. **The formal expression:**

   exists('security_initiatives')(system)
   ∧ exists('directives')(system('security_initiatives'))
   ∧ has_property('management_support')(system('security_initiatives'))

   Comments:

- *There must be a* document *named* 'security_initiatives',
- *there must be a* document *named* 'directives',
- *say, as a* sub-document, *in the* document, *d*, *named* 'security_initiatives', *and*
- *there must be a obvious, i.e., "visible" property of d*
- *namely that it has* 'management_support'.

5. **The formal expression:**

```
is_adequate(system('resources')),
                    (resources(system('information_security_policy')))
```

Comments:

- system('resources') *yields all* system resources.
- resources(system('information_security_policy')) *yields a "catalogue" of resources, say by name, needed to fulfill the* 'information_security_policy'.
- is_adequate*is a predicate.*
- *It applies to a catalogue of "real"* resources,*by value, and to a "catalogue" of resources, by name, and yields truth if the former are sufficient to satisfy the latter.*

6. **The formal expression:**

```
exists('role_assignment')(system('information_security'))
∧ exists('responsibilities')(system('information_security'))
```

Comments:

- approval *is here taken to be tantamount to the* existence *of the designated* assignments.

7. **The formal expression:**

```
is_aware('information_security')(system)
      ⊃ exists('plans')(system('information_security'))
           ∧ exists('programs')(system('information_security'))
```

Comments:

- is_aware *is a rather "sweeping" predicate.*
- *Its implementation is simple:*
  - * *one sends an e-mail to all staff to inquire "are you aware of plans and programs to maintain information security ?".*
  - * *If a significant percentage replies yes, then predicate yields true !*
- *More "formally"* awareness *implies that the designated* plans *and* programs *(documents and [probably] software) are found (somewhere) in the* system.

8. **The formal expression:**

   exists('information_security_controls')(system)
   $\supset$ is_coordinated('information_security_controls')(system)

   Comments:

   - *For this 'code of practice' we have, if not "given up" then at least (again) resorted to some rather "sweeping" generalisations:*
     * *First we have postulated that there is a* document*by the name* 'information_security_controls',
     * *and that that* document *does indeed address the issues covered by its name.*
     * *Then we have used the same name* ('information_security_controls') *as the name of a* concept
     * *and postulated an again "sweeping" predicate,* is_coordinated, *which "tests" the* system *for being in compliance with this* concept.
   - *The implementation of* is_coordinated *could be like that of* is_aware *above (Item 7 on the preceding page).*

[9.1.1] Physical security perimeter: .

- **The 'Code of Practice' Statement:**

  The following guidelines should be considered and implemented where appropriate for physical security perimeters:

  1. security perimeters should be clearly defined, and the siting and strength of each of the perimeters should depend on the security requirements of the assets within the perimeter and the results of a risk assessment.

- **A Predicate Term Interpretation:**

  1. **The informal expression:**

     security perimeters should be clearly defined, and the siting and strength of each of the perimeters should depend on the security requirements of the assets within the perimeter and the results of a risk assessment;

     The formal expression:

     is_well_defined('security perimeter')(system) $\wedge$
     let ra = risk_assessment(system), sr = security_requirements(system)
     sas = siting_and_strength(system) **in** is_commensurate((ra,sr),sas) **end**

     Comments:

- *An overall comment is this:*
  - *The informal 'code of practice' assumes quite a lot:*
    - *that there is a complete understanding of the physical plant, i.e., the land site, its borders to and bordering with other sites; the composition of buildings on this site; the one or more floors of each of these buildings; their floor plans; etc., etc.*
- ...

We leave the remaining items of the ISO document [9.1.1] Physical security perimeter further untreated.

5 The Phenomena of IT Systems

The observable, manifest phenomena are: *simple entities*[1], *functions, events and behaviours.* Besides phenomena, *"that which we can see, hear, touch, smell, and taste"* and (or) measure with physics- (incl. chemistry-) based instruments, there are concepts. We shall treat concepts later.

Our *treatment of phenomena and concepts* is in the form of rough sketches, that is, not systematic, as a narrative, and not formal — but will later be. Also, we shall not establish a proper terminology but ought to have. We leave that as an exercise to the reader.

5.1 Simple Entities

5.1.1 General

By a simple entity we shall understand something physical, something we can point to, something which occupies space, or something which is an abstraction, a concept, thereof. Simple entities might "end up", in a computing system, like data in a database, or data associated with variables in a program. Simple entities are the "things" to which we apply functions.

5.1.2 First Examples of Simple Entities

Examples of simple entities are (1) the fixed physical plant: (a-b-c-d-e-...) buildings: halls, stairwells, corridors, rooms, etc., and (f-g-h-i-...) the ground around buildings: roads, walkways, parking areas, etc., (1) the installable semi-fixed building parts: (a) electrical wiring, (b) water and sewage piper, (c) burglary alarm systems, (c) fire detection and fire extinguish systems, (..) etc.; (2) the installable and relocatable (IT security-related)

[1] We distinguish simple entities from entities. The later are all the phenomena and concepts of the domain. Functions apply to and yield entities; events involve predicates over entities. Since behaviours are sets of traces of actions and events — where actions derive from the application of operations to arguments — also behaviours involve the full notion of entities.

equipment: (a) main frame computers, (b) servers, (c)data communication cabling, (..) etc.; (3) the movable equipment: (a) mostly laptops; (4) people: (a) staff, (b) hired consultants, (c) clients, (d) potential customers, (e) invited visitors and (f) intruders; and (5) registers: (a) books and (b) databases (possibly kept on potentially movable storage media).

We shall now conceptually examine these simple entities more systematically.

5.1.3 Atomicity and Compositionality

One can can abstract a simple entity either as an atomic simple entity or as a composite simple entity. We decide to model a simple entity as an atomic simple entity if it is decided that it has not sub-structuring, that is, if one can not meaningfully, that is, in the context of the purpose of the model, decompose it into sub-entities. And we decide to model a simple entity as a composite simple entity if it is decided that it has a meaningful sub-structuring, hence consists of sub-entities. Atomic simple entities have attributes, that is, can be characterised by a number of properties, but these properties, as a whole, cannot be separated. Examples will follow. Composite simple entities have (i) simple sub-entities, (ii) a mereology, i.e., something which tells us how the simple entities are related to one another, and (iii) attributes. We shall consider these three kinds as independent of one another.

5.1.4 Atomic Simple Entities

An atomic simple entity is a simple entity whose possible "parts" we have decided not to consider, that is, to abstract from.

In one context a simple entity may be considered atomic while in another context it may be considered composite. In the context of IT Systems we decide to model human beings as atomic; while in the context of surgery (health care) we may decide to model human beings as composite.

Examples of Atomic Simple Entities. We give two examples of atomic entities of IT systems.

The first example of an atomic entity is that of a laptop. Its attributes are: brand name, model, serial number, storage hierarchy capacity, clock cycle, ports, etc.

The second example of an atomic simple entity is that of a human being. Personal attributes are: Name, gender, birth date, where born, citizenship, etc.; height, weight, color of eyes, etc.; education; IT skills; and IT responsibilities and IT authorisations.

5.1.5 Composite Simple Entities

A composite simple entity is a simple entity whose possible "parts" (that is, the sub-entities) we have decided consider, that is, to focus on — as well as how (the mereology of how) these simple sub-entities are put together. Add to our analysis of composite simple entities some properties that are properties of the composite entity, not of the simple sub-entities. We shall refer to these properties as attributes of the composite simple entity.

Simple Sub-entities and Their Mereology. Thus we shall examine sub-entities as "free-standing" components of composite entities, and we shall introduce the concept of mereology (the study and conceptual (philosophical) knowledge of "parts and wholes") to deal with the "free-standedness"!

Examples of Composite Simple Entities. We give two examples of composite simple entities.

The first example is of a building complex:

- Sub-entities of a building complex: the ground area of the building complex, the roads external to the ground area, the roads internal to the ground area, and the buildings on the ground area.
- Mereology of the simple sub-entities of a single floor building: Some external roads are connected to some internal roads, some buildings are connected to some internal roads, and some buildings are connected to some other buildings.
- Attributes of a building complex: the name of the building complex, the address of the building complex, the legal ownership of the building complex, the acreage (etc.) of the building complex, etcetera.

The second example is of a single floor building: the simple sub-entities are the entrance/exit ways of the building, the corridors and the rooms (walls, doors, windows, etc., are considered part of these entities); the mereology of a single floor building outlines the general or specific adjacency of entrance/exit ways, corridors and rooms; and the attributes are those of the name, owner(s), position (within some ground area), building materials, etc.

Attributes. We thus associate properties with atomic as well as composite simple entities. Simple entities have at least one attribute. We have decided that it makes no sense to speak of attribute-less simple entities.[2] We shall model an attribute as having a name (an attribute, or type name) and a value. A simple entity may have more than one attribute. In our narrative of multiply-attributed entities we do not consider their structuring (i.e., the "mereology"). We have concluded that any such perceived structuring of multiply-attributed entity attributes is irrelevant.[3]

[2] This is, of course, a conjecture. As such we are ready to one day admit its refutation. "Science only makes progress through refutations"!

[3] This is, of course, another conjecture. As such we are, also in this case, ready to one day admit its refutation.

Shared Attributes. We introduce a modelling notion of shared attributes. Examples are: a wall separating two rooms (or diving a larger room into two smaller rooms), a door (of a wall), being shared between two rooms and a window between a room and "an outside". A shared attribute may in one model not be modelled as a shared attribute, but as a sub-entity. An example could be a door (or a window) of a wall.

5.1.6 Summary of Simple Entities of IT Systems

We summarise simple entities of IT Systems 'of interests',[4] helter-skelter, with no apparent consideration of whether atomic or composite, or whether simple sub-entities of other simple entities: Next is a semi-structured, yet incomplete list of simple entities of IT Systems of interest: (1) *physical plant:* an or the IT System building complex, building ground, road, building, room, corridor, etc.; (2) *installations:* wiring, water piping, sewage piping, burglary detector & alarm, fire detector & alarm, fire extinguisher, etc.; (3) *movable equipment:* main frame, server, chair, table, cabinet, laptop, etc.; (4) *person*; and (5) *register*. You will have noticed, that we have grouped the IT System simple entities into five classes. This is a choice. We could have chosen another decomposition of simple entities into such classes. We shall later motivate the above grouping. The above choice will determine our formal modelling. Whether our choice is a good or a not so good choice will become apparent only if we formalise a number of alternative choices — and then evaluate their merits, their elegance.

5.1.7 Discussion

We will not in this document list "all" the simple entities of an IT System. Instead we will, in our formalisation introduce abstract, i.e., conceptual classes of simple entities. We have treated the analysis & modelling notion of IT System entities from an abstract, generic point of view, for example outlining composite phenomena of building complexes and buildings generically. In any particular application of the ideas of this document to a specific IT System the applier would then have to instantiate the general notion of building (etc.) mereologies to become very concrete. The above analysis & modelling approach applies to the next issues as well: functions, events and behaviours.

5.2 Functions

5.2.1 General

By a function[5] we shall understand *something*, an abstract concept, which when *applied* to a grouping of one or more entities, i.e. and *argument yields* a *result*, a value, in the form of either a grouping of *entities* or of *attributes* or a combination thereof.

[4] We single out the term 'of interest' to indicate that, in some other model of "basically the same domain", there could have been another choice of simple entities.

[5] We shall, inter alia, use the term 'operation' in lieu of the term 'function'.

5.2.2 Functions on (1) Physical Plant

Examples of functions that apply to entities of class physical plant are: (a) create a building, (b) change building attributes, (c) remove a building, (d) subdivide building rooms, (e) change wall attributes,[6] (f) connect two building, (g) create a road, (h) change a road, (i) remove a road, (..) etc.

5.2.3 Functions on (2) Installations

Examples of functions that apply to entities of class installations are: (a..) install wiring (piping, fire detector or alarm or extinguisher, burglary detector or alarm), (b..) change, reroute, wiring (etc.), (c..) remove wiring (etc.), (d..) change attributes of the above (wiring, piping, fire detector or alarm or extinguisher, burglary detector or alarm), (..) etc. All of the above are wrt. some sub-entities of some building, etc.

5.2.4 Functions on (3) Potentially Movable Equipment

Examples of functions that apply to entities of class potentially movable equipment are: (a) introduce (i.e., "create") such equipment, including placing it at some location, (b) moving mobile equipment from one location to another, (c) removing mobile equipment, (d) applying, for example a laptop or a main frame to a program, that is, invoking an IT Service, (e) changing attributes of mobile equipment, like installing, upgrading, or removing software or data, (..) etc.

5.2.5 Functions on (4) Persons

Examples of functions that apply to entities of class person are: (a..) hire, transfer, lay off or fire a staff, (b..) change attributes of a staff person: promote, demote, salary change, authorisation rights (privileges), etc., (c..) review or evaluate staff performance, (d..) allow a non-staff person to be admitted to a building or a room, or to perform some IT functions, etc., (..) etc.

5.2.6 Functions on (5) Registers

Examples of functions that apply to entities of class register are: (a) create a register, (b) update a register: (b.1) record the occurrence of a desirable or undesirable event, (b.2) evaluate a recorded event and so annotate the register, (b..) etc. (c) copy (part of) a register and (d) destroy a register.

5.2.7 Discussion I

As first presented above, functions are seen as mathematical abstractions. To apply a function to its arguments and obtain a result takes no time — time is not an issue. But in

[6]— like inserting a door, removing a door, changing the attributes of the door [access rights], etc.

a real world performing the kind of functions that are exemplified above does take time. And, as presented above functions are "functional", that is, they are not like procedures or subroutines of conventional, imperative programming languages like Java and C#, "our" functions do not act upon storage variables and change the values of these — they are mathematical functions. To prepare for a treatment of functions whose application takes time and may be understood as "altering" some input argument we now introduce a notion of state.

5.2.8 States

One may consider any composition of entities as a state. We usually make the pragmatic distinction between contexts and states. Contexts are compositions of simple entities whose value change less often and states are compositions of simple entities whose value change more often. Contexts provide a setting for activities, while states are the target of these activities.

5.2.9 State-changing Functions

We say that state-changing functions when invoked are actions. Actions may change the state and may "return" a value to the actor, see next, who invokes (triggers, ...) the function.

5.2.10 Discussion II

When functions are applied, then they are usually applied at some location, and at some time, by some actor, a person or a machine, or whose invocation is triggered by some event — we may say that some "outside" agent "is at play" — and maybe with some arguments provided by the actor who also designates the context and state entities on, or to which the function is to be applied.

So actors are either persons, or are machines, or are "outside" agents. We shall now treat the notion of events.

5.3 Events

5.3.1 General

Events "happen". They "occur". They take place instantaneously. They are like "communications" from an "outside". They are not functions — although they may, "mysteriously" trigger the invocation of functions; and they are not entities — although they may convey values. Later, when dealing with behaviours, we shall treat events as (synchronisations and) communications between behaviours — including the, or an, "external" behaviour. The notion of event is closely related to the notion of behaviour.

5.3.2 Examples of IT System Events

We give a number of examples of undesirable IT System events.

(1) *Events related to the physical plant:* (a) earthquakes, (b) typhoons, (c) fire, (d) break-in by un-authorised persons, (..) etc.

(2) *Events related to physical plant installations:* (a) electricity power break-down, (b) broken water pipes, (c) vandalism to communication cables, (d) break-down of fire detector and fire extinguisher, (e) break-down of burglary detection and alarm system, (..) etc.

(3) *Events related to potentially movable equipments:* (a) un-authorised access to a mainframe or laptop, (b) disappearance (theft or otherwise) of a laptop or a data medium, (c) sudden appearance in an unexpected place of a laptop, (..) etc.

(4) *Events related to persons:* (a) un-authorised access to a room (of a building) by some person, (b) un-authorised access to a mainframe or laptop by some person, (c) loss (theft or otherwise) of access entry card or password, (..) etc.

(5) *Events related to registers:* (a) the entries of a register are up for the annual review, (b) un-authorised access to (edit of, etc.) a register, (..) etc.

5.3.3 Event Identifier

By an event identifier we shall understand some unique way of identifying one set of events from another set. Examples of event identifiers: typhoon, earthquake, power break down, fire in building #A, water pipe breakage in building #B, etc.

5.3.4 Event Alphabet

By an event alphabet we shall understand a set of event identifiers. An example of an event alphabet is {typhoon, earthquake, power break down, fire in building #A, water pipe breakage in building #B, ...}

5.3.5 Synchronisation and Communication

We shall consider events as relating two (let us assume simple) behaviours where simple behaviours are seen as sequences of actions and events, in any order, and where events synchronise the progress of these two behaviours while possibly also communicating values between them.

5.3.6 Discussion

The above represent a greatly simplified notion of events (and behaviours). It will do for all of our present modelling. It is based on the process concept of CSP: Communicating Sequential Processes. Other notions of events and behaviours could have been used for example the Petri Net or the π-Calculus notion of processes.

5.4 Behaviours

5.4.1 General

By a behaviour we shall — somewhat circularly — understand a sequence of sets of actions and events.

5.4.2 Simple, Single-thread Behaviours

By a simple behaviour we shall understand a (linear) sequence of actions and events.

5.4.3 Composite, Multiple-thread Behaviours

By a composite behaviour we shall understand a set of simple or composite behaviours.

5.4.4 Communicating Behaviours

By a pair of communicating behaviours we shall understand two simple or composite behaviours such that an event in one of these two identifies an event in the other of these two.

5.4.5 Communications

Let

$$c_i :< a_{i_1}, ..., e_{ij}, ..., a_{i_m} >$$

and

$$c_j :< a_{j_1}, ..., e_{ij}, ..., a_{j_n} >$$

describe two behaviours ($\mathcal{C}_i$, $\mathcal{C}_j$). The a_{i_k}'s and a_{k_ℓ}'s describe actions ($\mathcal{A}_{i_k}, \mathcal{A}_{j_\ell}$) internal to $\mathcal{C}_i$, and $\mathcal{C}_j$, respectively. e_{ij} describes an event $\mathcal{E}_{ij}$. Since e_{ij} occurs in both c_i and c_j event $\mathcal{E}_{ij}$ may occur in both $\mathcal{C}_i$ and $\mathcal{C}_j$. If $\mathcal{E}_{ij}$ occurs in both $\mathcal{C}_i$ and $\mathcal{C}_j$, then it occurs simultaneously in both behaviours.

Internal Communications. Let k designate a channel, e an expression, v an identifier, and let e_{ij} be of the "paired" forms

in ci: k!e, in cj: **let** v = k? **in** ... **end**

then, when event $\mathcal{E}_{ij}$ occurs between behaviours $\mathcal{C}_i$, $\mathcal{C}_j$, the following happens: e is evaluated in $\mathcal{C}_i$, the value is bound to v in $\mathcal{C}_j$, and the two behaviours proceed. We say that the *two behaviours* have been *synchronised* and that *a value* has been *communicated* from one to the other. We say that the communication has been internal between the two behaviours.

External Communications. If either behaviour $\mathcal{C}_i$ or $\mathcal{C}_j$ has been left out of our description (i.e., c_i or c_j has not been given), then we say that the communication has been external between the described behaviour and an "external world".

5.4.6 Discussion

We have presented a capsule view of behaviours (and events). There is more, much more, to say, but this shall suffice. The view presented is basically that of Hoare's CSP: Communicating Sequential Processes. It is the CSP view of behaviours that we shall assume in the following.

5.5 Discussion

We have presented a view of entities, functions, events and behaviours. We take these four concepts as forming, one could say, one coherent set of aspects of an ontology of descriptions. We shall next take a brief look at other sets of aspects of an ontology of descriptions.

6 A Formal Model of IT Systems

6.1 Ω: The "Grand" State

In the modelling of all observable entity phenomena: simple entities, operations (i.e., functions), events and behaviours, we make use of the notion of a "grand state" $\omega : \Omega$. The grand state includes basically all observable simple entities. We name by Ω the type of all "grand states". We usually name by ω a value in Ω, i.e., a "grand state". Usually functions performed on, i.e., actions within the IT system being modelled are of either of the following signatures:

```
type
    Ω, ARG, VAL
value
    val_f: ARG → Ω → VAL
    int_f: ARG → Ω → Ω
    elab_f: ARG → Ω → Ω × VAL
```

That is, these functions are either **eval**uation functions observing, extracting or calculating (i.e., computing) a value of some ω, or **int**erpretation functions "changing", updating ω into ω', or are **elab**oration functions observing, extracting or calculating (i.e., computing) a value of some ω while "changing", updating ω — the latter is then called a "side-effect".

We model the grand state as consisting of several subsystems (one could call them components):

- Φ: the plant,
- Θ: the installations,
- $\Phi\Theta$: the plant and installations,
- Σ: movable equipment,
- Π: personnel and
- $\Re$: registers.

We now discuss these.

Formally we shall consider Ω to be a sort equipped with observers for at least each of the major sub-systems.

Since we shall be modelling the plant and the more-or-less fixed installations as one (highly structured "component") sub-system, of sort $\Phi\Theta$,

```
value
    obs_ΦΘ: Ω → ΦΘ
    obs_Σ: Ω → Σ
    obs_Π: Ω → Π
    obs_ℜ: Ω → ℜ
```

Predicates applicable to $\Phi\Theta$ can then be defined to discriminate between plant components (or sub-systems) and installations. The reason for modelling the two otherwise somewhat distinguished sub-systems is that the highly intricate structuring of installations (such as pipes, wires and cables) follows the similarly highly intricate structuring of the plant.

6.2 $\Phi\Theta$: The Plant and Installations

We shall develop our model of the plant + its installations by "slowly" unfolding a notion of system diagrams and system graphs. The system diagrams are very much like architectural drawings, i.e., building and floor plans, whereas system graphs are just that: graphs with nodes and edges. Nodes correspond to rooms (or an installation) of a building whereas edges correspond to access to rooms (i.e., a door or a barrier) or access to installations (a water pipe crane, an electric wire adaptor, a sewage pipe drainage, etc.).

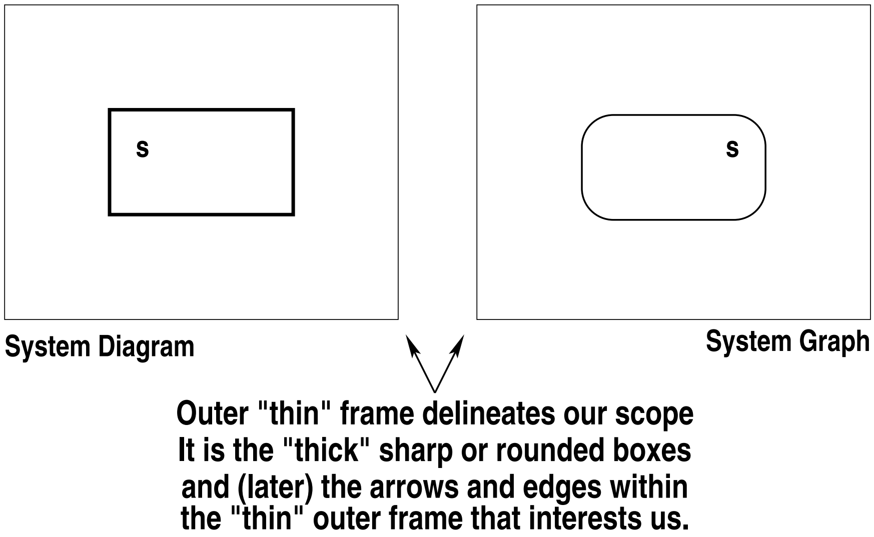
So our "pedantic unfolding" of how buildings are composed from rooms, and of how rooms may be considered "embedded" in "larger" rooms, or, rather, embedded in sub-parts of a building, e.g., floors or (east, center, north, etc.) wings of a building that pedantic development starts from basic, atomic entities and proceeds via their composition, to the general composition of composite entities (i.e., nodes) and the accesses from nodes (i.e., rooms, etc.) to nodes (i.e., adjacent rooms, etc.).

We develop the model for the plant + its installations by first developing two graphic languages: a language of system diagrams, and a language of system graphs. There are not many step in their development, but they are, as we have now said several times, a bit pedantic, so please bear with us.

6.2.1 Simple Composition Rules

1. **A simple atomic plant:**

   The simplest plant is one consisting of just one atomic component. See Fig. 1.



Outer "thin" frame delineates our scope
It is the "thick" sharp or rounded boxes
and (later) the arrows and edges within
the "thin" outer frame that interests us.

Figure 1: A simple atomic plant

   The sharp edged box (rectangle) in the system diagram is reminiscent of how one might draw a layout of a building, or a map of a collection of buildings (in this case only one), or a machine, or, for that matter a single human. The rounded corner box in the corresponding system graph is going to be our graphical notation for plants: a plant, a component, "is" a node.

2. **A simple composite, embedded plant:**

   The simplest composite plant reflecting embeddedness consists of one composite component, s, which then has one simple atomic component, s_e, embedded within s.

   Now we have a node within a node, as in hyper-graphs. Plant s appears not to be able to "access" sub-plant s_e — whatever we mean by 'access'. (We will elaborate on that later, but you can think of access as meaning: for a properly authorised human to "use" a plant, being able to perform the (one or more) function(s) that the plant may offer, being able to read, update, copy, etc., the information that the plant "embodies" or the functions it offer.)

3. **A simple composite, embedded plant with access:**

   The simple embedded plant of Item 2 did not show the possibility of accessing subplant s_e from plant s. We modify Fig. 2 on the next page into Fig. 3 on page 40[A].

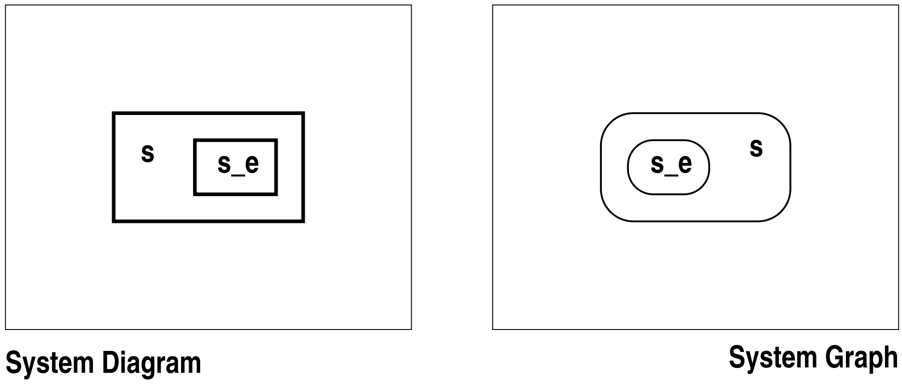


Figure 2: A simple composite, embedded plant

We have in the system diagram of Fig. 3 on the next page[A] (left) shown an "arrow" (mostly, really, an arrow-head) to indicate that one can "access" embedded component from "outer" components. The access is suggested to be directional, one way, in one direction, or in the opposite direction, or two-way, in both directions. The system diagram "arrow" is "dangling": it is not shown from where "within" plant s the arrow emanates and it is not shown to where "within" sub-plant s_e the arrow is incident. In the system graph of Fig. 3 on the following page[A] (right) we show the "dangling arrow" notation of Fig. 3 on the next page[A] (left): the system graph edges from nodes to sub-nodes are dashed.

In Fig. 3 on the following page[B] we show three possibilities of access.

4. **A simple composite, disjoint plant:**

   The simplest composite, non-embedded plant has the plant s consist of two adjacent, that is, two disjoint sub-plants si and sj. See Fig. 4 on page 41.

   Sub-plants si and sj appear not be accessible from plant s and it also appears that one cannot access either of the sub-plants from the other.

5. **A simple composite, adjacent plant:**

   We can juxtapose two disjoint sub-plants si and sj "right" next to one another, that is, adjacently, "sharing" some "wall". See Fig. 5 on page 41.

   We shall soon see what that 'wall' means, that is, makes possible. As it stands now, in Fig. 5 on page 41, there seems not to be access between the two sub-plants. Note the straight line between nodes si and sj of the system graph. It models the wall, i.e., adjacency (not access).

6. **A simple composite, disjoint and adjacent plant:**

   We "insert" some access arrows in the wall of Fig. 5 on page 41 to contain Fig. 6 on page 42.

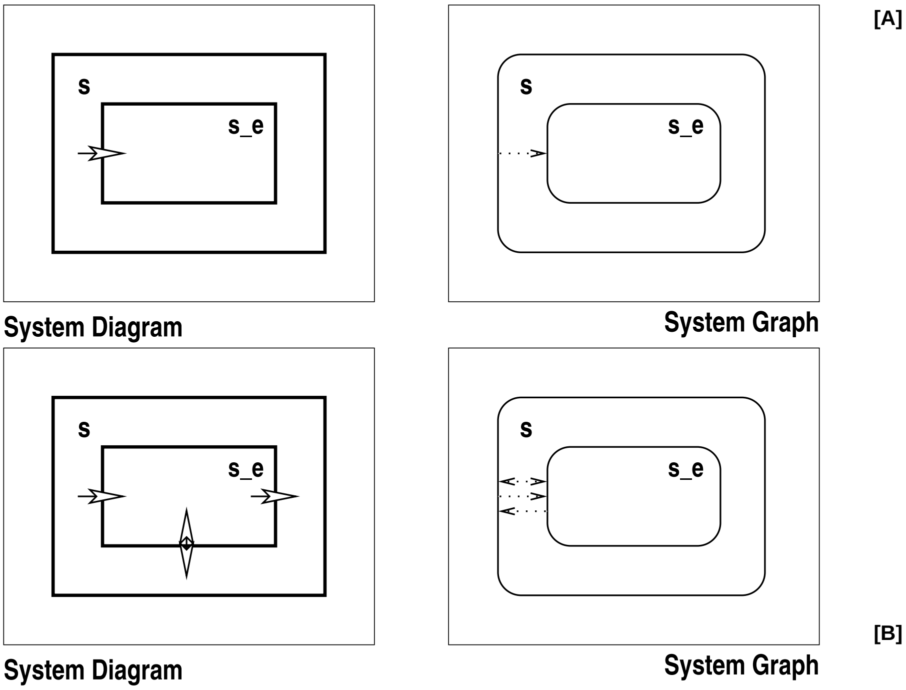


Figure 3: A simple composite, embedded plant: [A] one access; [B] three accesses

The meaning is that now si and sj can access one another. We need only have shown one access arrow: either one-way from si to sj, or two-way sj between si, or one-way from sj to si — as shown, top-to-bottom in Fig. 6 on page 42. The (three) un-dotted (i.e., straight line) arrows of the system graph designate both adjacency and access direction.

7. **Embedded Adjacent Sub-plants with Access:**

   Let us consider, see Fig. 7[A], a sub-plant s_{i_j} of a sub-plant s_i of plant s such that "activities" of s can directly access the "inner" sub-plant s_{i_j}. In the system diagram we show the sub-plant s_{i_j} "sharing" a wall" with sub-plant s_i, i.e., a wall between s and the two sub-plants (one, s_{i_j}, "within" the other, s_i).

   In the system graph of Fig. 7[A] we show this not by "sharing" the contour of s_{i_j} with that of s_i but by a dash-dotted line from the contour of s through the contour of s_i to the contour of s_{i_j}.

   Choosing this graphical rendition disambiguates any possible multiple interpretations as to which level of embedded sub-plants are being "connected". See Fig. 7[B].

We have introduced the most basic rules for composing plants: embedding and juxtaposition. We have shown how one can transform a system diagram of boxes into a system graph of nodes. And we have introduced the most basic rules for designating access, that is, for composing (system diagram) component boxes and (system diagram) access arrows.

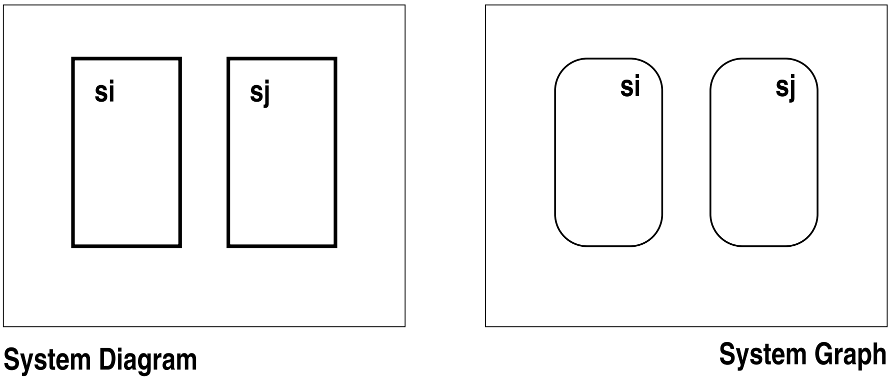


Figure 4: A simple composite, disjoint plant

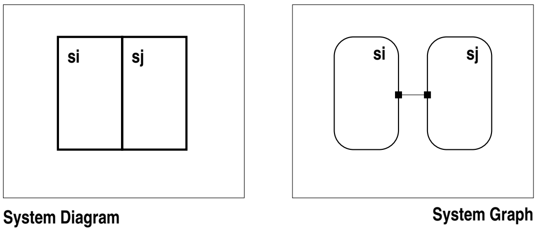


Figure 5: A simple composite, adjacent plant

6.2.2 Generality of the Simple Composition Rules

There can be any number m of sub-plants $s_{e_1}, s_{e_2}, \ldots, s_{e_m}$ embedded in a plant s, and there can be any number of juxtaposed (i.e., adjacent) sub-plants $s_{a1}, s_{a2}, \ldots, s_{am}$ in a plant s. Finally there can be any number of accesses (i.e., access arrows) between a plant s and an embedded sub-plant s_i of s and between any two adjacent plants s_{ai} and s_{aj} — even multiple occurrence of the same kind. What that means we shall cover later.

6.2.3 Composite (Combined) Composition Rules

We now analyse combinations of embedding, juxtaposing and access.

8. **Access between embedded sub-plants of adjacent plants:**

   Let si and sj be two disjoint, but adjacent plants. See system diagram of Fig. 8 on page 44. Let plant si_a be a sub-plant of si, and let si_{a_p} be a sub-plant of si_a. Similarly for sub-plant sj_x of sj. The system diagram of Fig. 8 on page 44 now illustrates all possible (in this case two-way) accesses between the two adjacent plants and all their respective sub-sub-plant. (Figure 8 on page 44 does not illustrate accesses from "outer" plants to embedded sub-plants of neither si nor sj. This is left as an exercise

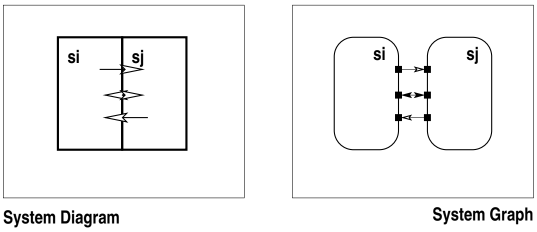


Figure 6: A simple composite, disjoint and adjacent plant with access

for the reader to draw: Both the system diagram and the corresponding system graph.)

Note that the topmost edge from plant *si* to disjoint, but adjacent plant *sj* is a solid line two-way arrow. All other edges are "dash-dot" (− · − · − · −) two-way arrows. By an access path, a route, we mean a direct access that involves "transgressing" zero, one or more "walls", between plants. All of the above accesses are composite. We can model an access path as follows:

type
  AP = S × S
examples:
  (si,sj), (sj,si), (si,sj_x), (sj_x,si), (sj,si_a), (sj_x,si_a_p)

Humans "transgress" access paths. Sometimes "transporting" plants. Each "transgression" amount to performing some function on the access.

9. **Access Routes:**

   By an access route, r, we mean a sequence of one or more access paths such that if p_{k-1}, p_k is a pair of "adjacent" paths in r then the second state (s_i) of p_{k-1} is the same as the first state (s_j) of p_k, that is, rewriting r:

   r: ⟨(s_1,s_2),(s_2,s_3),...,(s_j,s_j+1),(s_j+1,s_j_+2),...,(s_m−1,s_m)⟩

   See Fig. 9 on page 44.

   The system diagram of Fig. 9 on page 44 indicates the *route* while the system graph indicates the number of times the routes meanders its way through accesses (access points). Humans "travel" access routes. Sometimes "transporting" plants. 'Travelling' amounts to performing a sequence of functions on respective accesses.

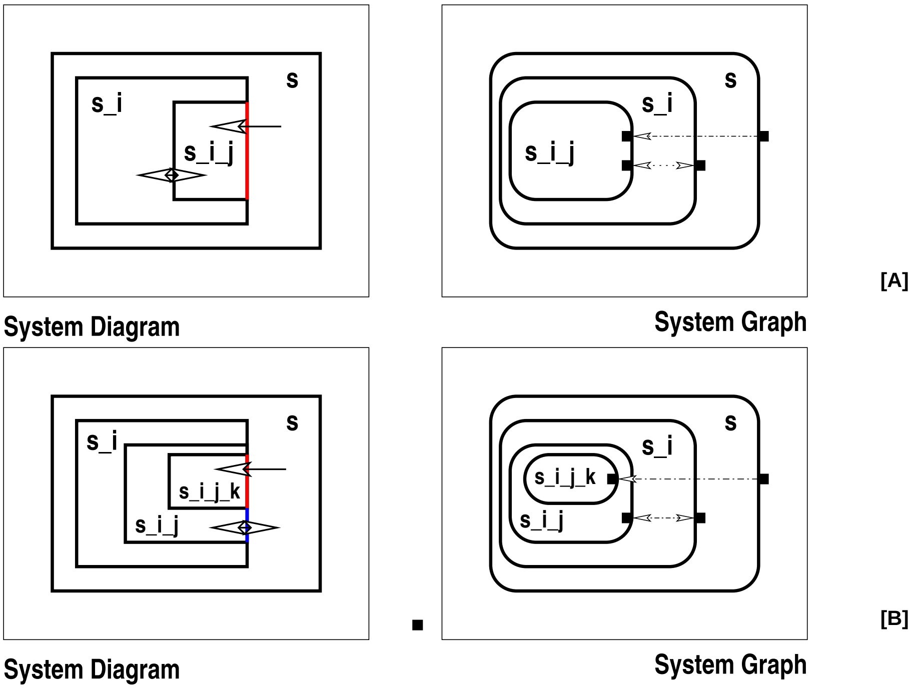


Figure 7: [A] Doubly embedded plant, triply embedded plant [B]

6.2.4 Planar and Non-planar Graphs

You may have noticed that all our system graphs were shown as planar graphs. You may also have wondered about the two-dimensionality of our system diagrams. The plants that we deal with in humanly manifest physical plants, that is, plants of roads, terrain around buildings, buildings and their internal layout, equipment within buildings, the possible electrical of electronic (wired or wireless) communication "cabling", etc., these plants and sub-plants are all three dimensional.

Is there a fourth dimension, or are there more than four dimensions? Is time a dimension? If the plants change their configurations of disjointness, adjacency or embeddedness, or if access paths change, is that something that is modelled in the time domain? We shall look at some of these issues now, and eventually at more of them. (Is it possible to eventually state that we have considered all such "dimensionality" issues?)

10. N **Adjacent Embedded Plants:**

    Consider an n story building, floor stacked upon floor. Usually a staircase connects the floors. A system diagram would then show the building as the plant and the staircase plus n floors as $n+1$ sub-plants. To get (i.e., to "access") from one floor to another one would have to pass through two accesses, each access being between a floor and the staircase. We leave the design of the system diagram and the system graph as an exercise. Consider instead a building where for every floor there is a

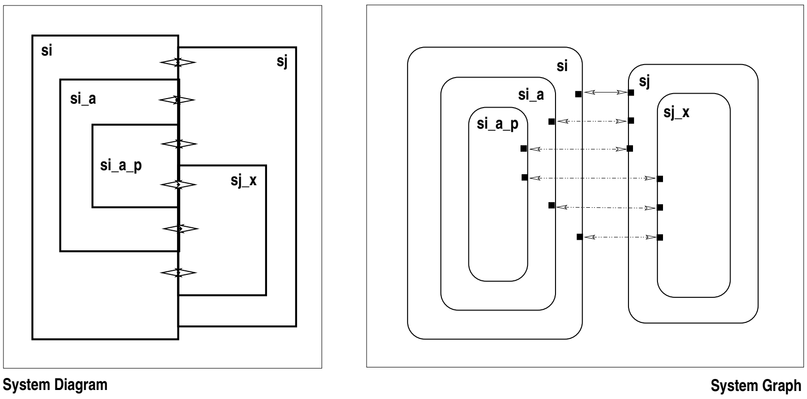


Figure 8: Access paths

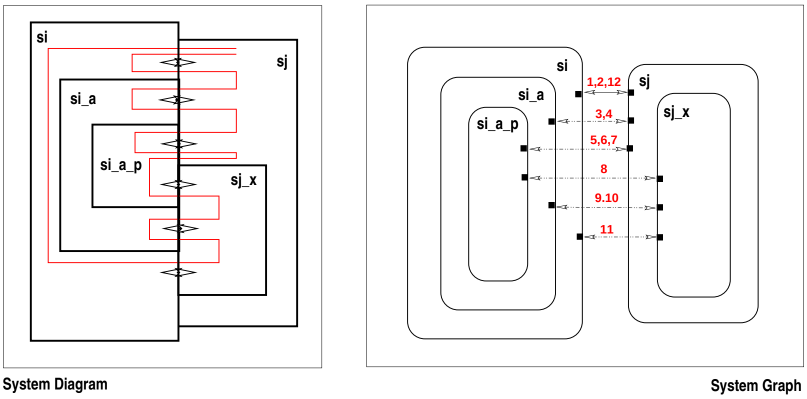


Figure 9: An access *route*

"bay" with a staircase to all the other floors such that only one access (one door) is necessary between any distinct pair of floors.

The system diagram considers the building as "separate" from the floors and considers the floor as disjoint sub-plants with only floor # 1 being adjacent to the building (i.e., its entrance hall).

The above construction shows that any three dimensional plant s can have any arbitrary number n of embedded sub-plants s_i of the plant be adjacent sub-plants. The two dimensional system diagram is inductive, cf. the use of "overlapping" floors and induction (...), hence it is schematic. Let us say that each horisontal floor plan is along dimensions X and Y, and that a vertical cut, along a vertical axis Z, through the building is along either dimensions X and ZS or Y and Z. Such a set of architectural plans or a proper isometric or perspective drawing of the building (or a set of such drawings together with floor plan drawings and a proper interpretation of those ensembles of drawings, would perhaps be the

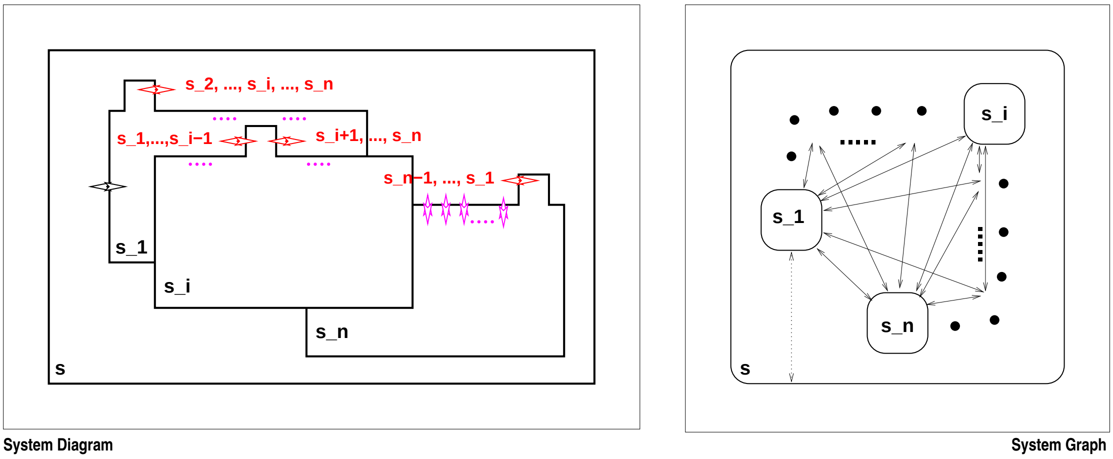


Figure 10: n Adjacent embedded plants

more proper way to show a three dimensional system diagram. There are similarly special diagrammatic languages for cabling (wiring), mechanical assembly, etc.

6.2.5 Conjecture

The essence of it all is that we can always map such three dimensional system diagrams onto a two dimensional system graph (albeit most often not a planar graph).

6.2.6 Examples of Plant Modelling

11. **Power supply cabling of machines:** See Fig. 11 on the following page.

6.3 A Formal Model of $\Phi\Theta$

6.3.1 The Syntax

12. Nodes have simple names (further undefined), and atomic (basic) components are further undefined.

13. A system graph G has a name, n:N, and otherwise consists of a basis part, b:B, a set of zero, one or more (disjoint) components (nodes), cs:C-**set**, and a set of zero, one or more edges, es:E-**set**.

14. A component, c:C, has a name, and consists of a basis part and a set of zero, one or mode components (ci) embedded in the defining component (c).

15. An edge connects two nodes, n1,n2:N, and has a set of one or more distinct access specifications, a:A.

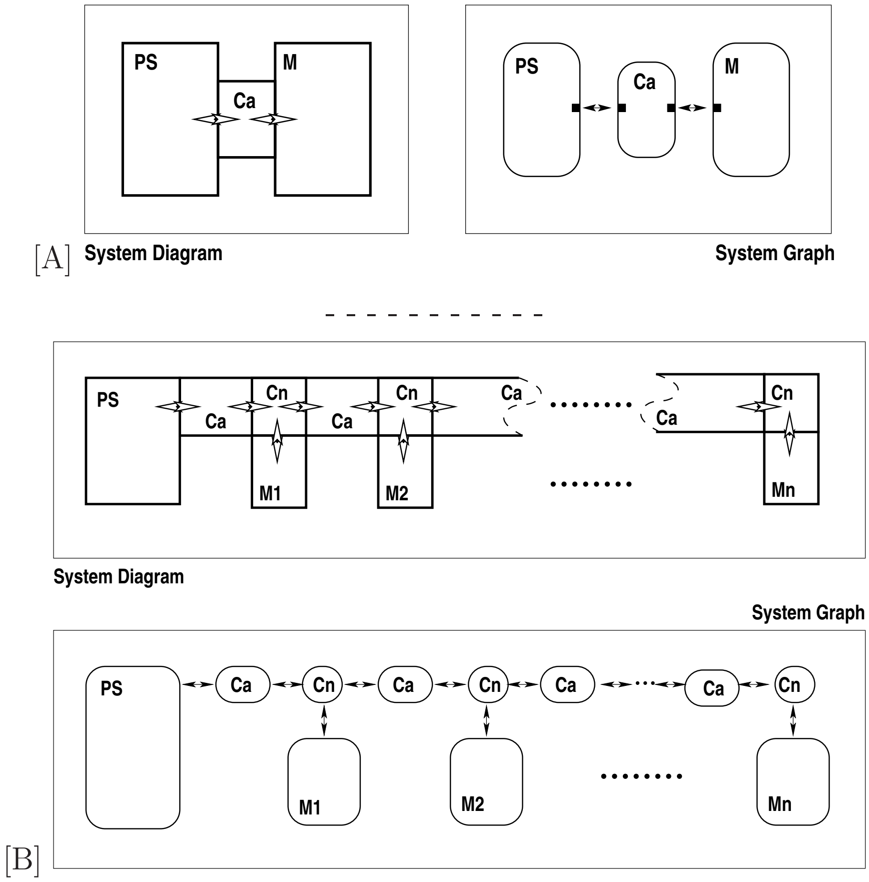


Figure 11: Power supply cabling of machines: [A] One machine, [B] n Machines

16. An access specification identifies an access direction and a set of operations.

17. A direction is either from the first to the second node (n1,n2), or the reverse, or is two-way. If the direction is adj_wall then there is no access but the two nodes possess an adjacent wall (and the operation set is empty).

18. An operation is either a move, or a read, or a perform, or some other operation.

19. These operations are left further undefined.

```
type
12  N, B
13  G == mkG(sn:N,sb:B,cs:C-set,es:E-set)
14  C == mkC(sn:N,sb:B,scs:C-set)
15  E == mkE(s1:N,s2:N,sks:A-set)
16  A == mkA(sd:D,sas:O-set)
17  D == fst_snd | snd_fst | 2_way | adj_wall
```

18 O == Move | Read | Perform | ...
19 Move, Read, Perform, ...

6.3.2 The Syntactical Constraints

12. All nodes of a graph, whether embedded or juxtaposed, have distinct names.

12,14. To paraphrase the above: Any two disjoint components, si and sj, of the components $\{s1, s2, \ldots, si, \ldots, sj, \ldots, sm\}$ of a plant s have distinct names and these are distinct from the name of s. Any component si is embedded in s and any two components si and sj, are disjoint (within s).

14. A component, i.e., a plant (or sub-plant — which is the same), $si : S$, has a name, n_{si}.

14. If a plant, s of name n_s, consists of only one component, $s1 : S$, of name n_{s1}, then their names, n_s and n_{s1} will be made be different).

20. We decide to secure distinct of nodes by mandating that names, n_{i_1}, n_{i_2}, $\ldots$, n_{i_m}, of nodes of immediate sub-plants of a plant named n_i are distinct and that the name n_i can be uniquely "extracted" from each n_{i_j} for all j in the interval $1..m$.

20. That is, think of the immediate components, s_{i_1}, s_{i_2}, $\ldots$, s_{i_m}, of s as being ordered as just listed, and the names being a bijection function, η of the name of the plant and the name index of the sub-plant:

type
  Idx
value
  η: N × Idx $\leftrightarrow$ N
  η^{-1}: N $\leftrightarrow$ N × Idx
axiom
  $\forall$ n:N, i:Idx • $\eta_\circ^{-1}\eta$(n,i) $\equiv$ (n,i), i.e.: $\eta^{-1}\circ\eta \equiv$ λx.x $\equiv \eta\circ\eta^{-1}$

15. An edge connects two nodes, n_α, and n_β. These nodes must be distinct. The two nodes stand in either of the following relations to one another:

(a) Either they are of disjoint but (of course) adjacent plants (otherwise why have the edge unless to express adjacency?),

(b) or one node is of a sub-plant embedded one or more levels within another (the "outer, surrounding") plant,

(c) or they are sub-plant nodes, n_α, n_β, each embedded (one or more levels, i.e., $\ell\#a$, $\ell\#b$), within disjoint and adjacent plant n_i, n_j. The α, β indexes typically would be: $i_{i1_{i2\ldots i\ell\#a}}$ respectively $j_{j1_{j2\ldots j\ell\#b}}$. (The number of $\ldots$ in these past two index expressions are $\ell\#a - 3$, respectively $\ell\#b - 3$.)

6.3.3 Formalised Graph Well-formedness

```
value
  wf_G: G → Bool
  wf_G(mkG(n,_,cs,es)) ≡
    let ns = all_nodes(cs) in wf_Cs(n,cs) ∧ wf_Es({n}∪ ns,es) end

  wf_Cs: N × C-set → Bool
  wf_Cs(n,cs)
    let ns = {sn(c)|c:C•c ∈ cs} in
    let ixs={i|i:Idx,n′:N•n′ ∈ ns∧let (n″,j):(N×Idx)•η⁻¹(n′) in i=j end} in
    card cs = card ns = card ixs ∧ n ∉ ns ∧
    ∀ mkC(n′,_,cs′):C • mkC(n′,_,cs′) ∈ cs ⇒ wf_Cs(n′,cs′) end end

  wf_Es: N-set × E-set → Bool
  wf_Es(ns,es) ≡ ∀ mkE(n,n′,_):E • mkE(n,n′,_) ∈ es ⇒ {n,n′}⊆ns
```

6.3.4 Mereological Operations on ΦΘ

By a syntactic operation on a plant we mean an operation which changes its hypergraph representation. Humans perform such operations. Some operations on certain components or entities require authorisation.

21. Plants change dynamically.

22. One may

    (a) **adjoin a node to a plant** with the new node being disjoint to all other nodes of the plant,

    (b) **embed a node in a plant**, with the new node being immediately contained in some node of the plant,

    (c) **connect two nodes**, whether disjoint or arbitrarily contained.

    (d) **sever, i.e., remove, the edge between two nodes**, whether disjoint or arbitrarily contained.

    Etcetera.

We leave it as an exercise to formalise these operations.

6.3.5 Attribute Operations on $\Phi\Theta$

By an attribute operation on a plant we mean an operation which changes changes the attributes associated with nodes and edges. Humans (and foreseeable or unforeseen non-human events) perform such operations. Some operations on certain components or entities require authorisation.
We leave it as an exercise to conceive of and narrate and formalise such operations.

6.3.6 Semantic Operations on $\Phi\Theta$

By a semantics operation on a plant we mean an operation which invokes a function to be applied to the plant. Humans (and foreseeable or unforeseen non-human events) perform such operations. Some operations on certain components or entities require authorisation.
We leave it as an exercise to conceive of and narrate and formalise such operations.

6.4 $\Sigma\Pi\Re$: Movable Resources

6.5 Simple Entities

The movable resources, to recall, include:

- laptops,
- people and
- registers.

6.5.1 Modelling

We suggest to model a movable resource as a node in the system graph. At any one point two or more such nodes may be connected, i.e., are accessing one another, and/or are accessing nodes of the fixed (the $\phi\theta$ graph).

6.5.2 Operations

With movable resources we can associate a number of operations: (i) on laptops: (i.1–.2) (de-)register, (i.3) move, (i.4) query, (i.5–.6) (dis-)connect and (i.7) use; (ii) on people: (ii.1–.2) (de-)register, (ii.3) query, (ii.4–.5) (re) assign authorisations (ii.6) move, and (iii) on registers: (iii.1) (de-)catalog, (iii.2) update, (iii.3) query, (iii.4) search, and (iii.5) copy.

We leave it as an exercise to narrate and formalise these and other operations.

6.6 Discussion

We have sketched how one may narrate and formalise the simple entities and operations of an IT system. We have, and shall not cover the issue of events and behaviours of an IT system. But once a model of simple entities and operations is being established — true to scale, that is, with all the “bells & whistles” — one can then also model events and behaviours.

The simple entities, the operations, the events and the behaviours are all identifiable.

Their names must reflect the names used in the formal predicates of Sect. 4 (on Pages 24–25). These names identify IT System resources that are spatially related to one another as reflected in the ΦΘ and ΣΠℜ. Each identifier of any phenomenon or concept can thus be mapped into its "position" in the system graph as modelled in Sect. 6.

7 A Formal Modal of IT Security Code of Practice

Very preliminary remarks: We model the "code[s] of practice" as well-formed formula (*wff*) in a first order predicate calculus. The ground (mostly non-Boolean valued) terms denote entities in Ω. Predicate symbols denote predicates as we identified them in the logical explication of the code[s] of practice. Function symbols denote functions as we identified them in the logical explication of the code[s] of practice. Evaluation of a *wff* now take place in the context of some $\omega \in \Omega$.

7.1 $^{\Psi}$Syntax

We claim that the formal expressions of Sect. 4 on page 23 can all be expressed as well-formed formulas (*wff*s) in a predicate calculus. Below we present an (example annotated) abstract syntax for WWFs.

Since this is standard knowledge we make no further comments at this place, but refer to Sect. 9.5.5 (pages 178–180) of .

```
type                                          examples
   Cn, Vn, Pn, Fn, Tn                            cn, vn, fn, pn
   Term = TId | TAp
   TId  :: Vn | Cn                               cn, vn
   TAp  :: (Fn|Pn) Term*                         pn(t1,t2,...,tm), fn(t1,t2,...,tm)
   Atom = Aid | AAp
   AId  :: Vn | true | false                     vn, true, false
   AAp  :: Pn Term*                              pn(t1,t2,...,tm)

   wwf:WWF  = Atom|NWff|AWff|OWff|IWff|EWff|QWff

   NWff :: WFF                                   ∼wff
   AWff :: WFF WFF                               wff ∧ wff′
   OWff :: WFF WFF                               wff ∨ wff′
   IWff :: WFF WFF                               wff ⇒ wff′
   EWff :: WFF WFF                               wff = wff′
   QWff :: Quan Vn Tn WFF                        ∀ wff, ∃ wff′
   Quan == all | exist
```

7.2 $^{\Psi}$Semantics

By the semantics of a language, WFF, of *wff*s we mean an interpretation of the *wff*s in some context. The context assigns meaning to all symbols: The meaning of a predicate symbol is a predicate function of an arity commensurate with the number of terms following the predicate symbol. The meaning of a function symbol is a function of an arity commensurate with the number of terms following the function symbol. The meaning of a variable name is given by its typed binding in a quantified expression. The meaning of a constant name is given by the instantiation of a given plant (i.e., by some ω). The meaning of a type name is the set of all values of that type. And so forth.

All this is standard knowledge we make no further comments at this place, but refer to Sect. 9.5.7 (pages 181–184) of .

There is, however, a small technicality. It has to do with the context in which the *wff*s are interpreted. We normally see this context as a map from constant and variable identifiers, predicate and function symbols, etc., to their meaning. So, from the instantiated ω of the IT system being studied we prepare a context which maps all possible component and access (edge) names to their meaning (the designated physical artifact, including person, or the concept identified) — this was, amongst others, a reason for insisting on unique component and access names. The predicate and function symbols *wff*s of Sect. 4 on page 23 are likewise bound in an initial context to their meaning. Pls. observe that some of these predicate and function symbols may not denote computable functions — so we treat them as oracles.

7.2.1 The Context

```
type
  $i\Omega = (Cn|Vn|Pn|Fn|...) \underset{m}{\rightarrow} VAL$
  $VAL = (VAL^* \overset{\sim}{\rightarrow} VAL)$ | Bool | Int | ...
value
  $c\omega: \Omega \overset{\sim}{\rightarrow} i\Omega$
```

7.2.2 The Meaning Functions

```
value
  M: WFF → iΩ → Bool
  M(wff)iω ≡
    case wff of
      $mkNWff(wff') \rightarrow \sim M(wff')i\omega$,
      $mkAWff(wff',wff'') \rightarrow M(wff')i\omega \wedge M(wff'')i\omega$,
      $mkOWff(wff',wff'') \rightarrow M(wff')i\omega \vee M(wff'')i\omega$,
      $mkIWff(wff',wff'') \rightarrow M(wff')i\omega \Rightarrow M(wff'')i\omega$,
      $mkEWff(wff',wff'') \rightarrow M(wff')i\omega = M(wff'')i\omega$,
      $mkQWff(all,v,t,wff'') \rightarrow \forall\ u \in i\omega(t) \bullet M(wff'')(i\omega \dagger [\,v \mapsto u\,]$,
```

$\text{mkQWff}(\text{exist},\text{v},\text{t},\text{wff}'') \to \exists\ \text{u} \in \text{i}\omega(\text{t}) \bullet \text{M}(\text{wff}'')(\text{i}\omega \dagger [\,\text{v}\mapsto\text{u}\,],$
$\_ \to \text{A}(\text{wff})\text{i}\omega$

$\text{A: Atom} \to \text{i}\Omega \to \textbf{Bool}$
$\text{A}(\text{mkAId}(\text{v}))\text{i}\omega \equiv \text{i}\omega(\text{v})$
$\text{A}(\text{mkAId}(\textbf{true}))\text{i}\omega \equiv \textbf{true}$
$\text{A}(\text{mkAId}(\textbf{false}))\text{i}\omega \equiv \textbf{false}$

$\text{A}(\text{mkAAp}(\text{nm},\text{lt}))\text{i}\omega \equiv \text{i}\omega(\text{pn})(\langle \text{V}(\text{lt}(\text{i}))\text{i}\omega | \text{i}\ \textbf{in}\ \text{lt}\rangle)$

$\text{V: Term} \to \text{i}\Omega \to \text{VAL}$
...

The definition of the Term eValuation function follows, as do the predicate and function symbol meanings, from the instantiated ω under study.

8 Making Use of The Formalisations

We have three formalisations:

- A formalisation of the IT System "Code of Practice" predicates of Sect. 4;
- a formalisation of IT Systems, Sect. 6; and
- a formalisation of "Code of Practice" predicate evaluation, Sect. 7

Now, how are we going to bring these three formalisations together.

- First we have to complete the formalisation of the IT System "Code of Practice" predicates, Sect. 4.
- Then we have to complete the formalisation of IT Systems, Sect. 6: that is, to define all the term functions and the auxiliar predicate used, but not otherwise defined in Sect. 4.
- The former have to be instantiated to a given, specific IT System. Section 8.1 will sketch how.
- The specific (client) IT System-instantiated predicates now serve as input to the predicate evaluator of Sect. 7. That "input" process will be discussed in Sect. 8.2.

8.1 Instantiating IT Security Predicates for Evaluation

The reader will have noticed that the predicates presented in Sect. 4 were generic: that is, applied to any IT System. The reader will also have noticed that there did not appear to be any explicit universal or existensial quantifiers in the predicates presented in Sect. 4. Some of the semantic functions of Sect. 7 were fully defined, notably the overall predicate evaluator, M, but the term evaluator, V, was not. To define that function would require that we first completed the formalisation of the IT System "Code of Practice" predicates, Sect. 4, identifying all the auxiliary functions whose arguments specifically designated fixed and movable resourses and that we also complete the formalisation of IT Systems, Sect. 6. Many of the predicates of Sect. 4, as was commented, are not computable, and many of the auxiliary functions will, most likely turn out to be not only non-deterministic but also necessarily underfined ("loosely defined").

Thus there is a "hidden" universal quantifier that ranges over all IT Systems, and, thus, for each of these, is bound to a specifically instantiated ω_ι and, hence, to a specific $i\omega_\iota$. Now, each of the identified constant and variable names, $cn : CN, vn : VN$, ranges over that which they are intended to denote: specific physical facilities: plant, installations and movable resources. Thus the "hidden" universal or existensial quantifiers must be made specific and their range must be enunciated for every specific embedding. That is: any one instance of a predicate applies to a (cn or vn) value of $i\omega_\iota$. Thus the IT System "Code of Practice" predicates of Sect. 4, shall have to be instantiated for all resources immediately contained in that "layer" ("embedding"), k, of $i\omega$. That value may, for example, be a room (i.e., a plant, etc.) which has embedded rooms (plants, etc.). Thus the IT System "Code of Practice" predicates of Sect. 4, shall have to be instantiated for these, layer $k+1$ embedded facilities, fixed or movable if applicable. And so forth.

8.2 Evaluation

8.2.1 Testing for IT Security Dynamically

Thus we can define a function M and apply it to any instantiated state ω_ι:

$$M(\mathit{wff}_\iota)(\omega_\iota)$$

where wff_ι is any conjugated ($\wedge$) and instantiated subset of "code[s] of practice". If the resulting value is `ff` the instantiated subset "code[s] of practice" have been violated. If the resulting value is `tt` the instantiated subset "code[s] of practice" have not been violated.

8.2.2 Testing for IT Security Statically

If we evaluate

$$M(\mathit{wff}_\iota)(\omega_\iota)$$

for any (valid) instantiated ω_ι then we are, in a sense testing whether the given set of instantiated wff_ιs constitutes a relative complete and consistent "code of practice".

8.2.3 A Caveat

Of course: we cannot mechanically evaluate $\mathrm{M}(\mathit{wff}_1)(\omega_1)$. Most of the predicates and term functions mentioned in a formalisation of the ISO Code of Practice are not computable. So what do we do ?

8.2.4 Interactive Evaluation

One can devise the evaluation process such that whenever the evaluator encounters a partially defined function then the process interacts with IT System Security stakeholders present the state of evaluation to them and requests their advice as to which course the ongoing evaluation should take. This interactive evaluation process can be refined to allow for "search trees" of evaluation: that is, the interactive evaluator keeps track of non-deterministic and multiple input choices, and pursues these in turn.

8.2.5 Conformance

How do we know that the ISO Code of Practice instantiation is commensurate with the un-instantiated version of the ISO Code of Practice axioms ? Well, since the un-instantiated, formal version is not computable we shall have to prove the correctness of the refinement, i.e., the instantiations. And that proof cannot be mechanised. It is a classical mathematical (logic) proof: a lot of brain-power and a lot of writing.

9 Closing

9.1 What is IT Security ?

The International ISO/IEC Standard 17799: Information technology: security techniques — code of practice for information security management does not provide any briefer characterisation of what is meant by IT System security than its approximately 135 pages of detailed, operational description.

This may be acceptable. An IT system is a very operational "affair". It embodies few abstractions. What we would like to see in the direction of a characterisation of "what IT System Security" is, is illustrated next.

9.1.1 When is an IT System Secure ?

An IT System is secure when an un-authorised user, after periods of trying to "enter" the system (1) *cannot find out what it is doing (i.e., protecting),* (2) *cannot find out how it is doing (whatever it is doing),* (3) *and does not know that she, the user, does not know* (1–2) *!*

The third part is introspective[7] wrt. the first two parts.

[7] cf. introspective logic of belief ...

This “definition” is, of course, highly debatable. But it makes a point: namely that one cannot pursue the issue of IT System Security without having a proper, not too long, and certainly not an approximately 135 page long, and definitely not an implicit “definition” such as the ISO/IEC Standard 17799. One must do far better than that. Whether our definition is a feasible one, or, with some preamble definitions could be made feasible we shall leave as an open question

9.2 What Have We Achieved?

We have “achieved” the following: (i) indicated, while providing formal “arguments” for, how IT System Codes of Practice rules could be formalised; (ii) indicated, while providing formal “arguments” for, how the context in which these IT System Codes of Practice rules must be understood; and (iii) indicated, while providing formal “arguments” for, how the [(i)] rules can be interpreted in their contexts [(ii)].

9.3 Issues of Contention

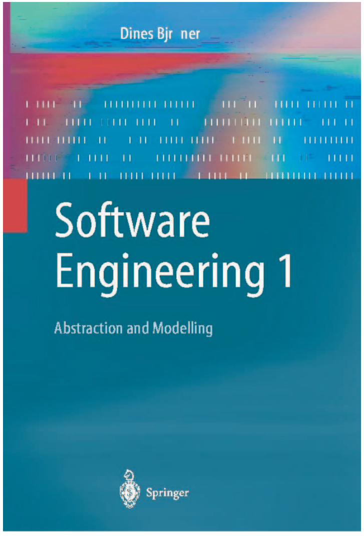
But the “formalised indications” (of Sect. 9.2) are merely sketches. And not all issues have been resolved. There are many “dangling”, i.e., unresolved issues: (i) completion of formalisation of IT System Codes of Practice; (ii) completion of formalisation of IT System facilities and resources; (iii) completion of semantic interpretation functions; and (iv) clarification of exactly how the free identifiers in the formalisation of the predicates and the auxiliary term functions of (i) shall be bound to the entities of (ii).

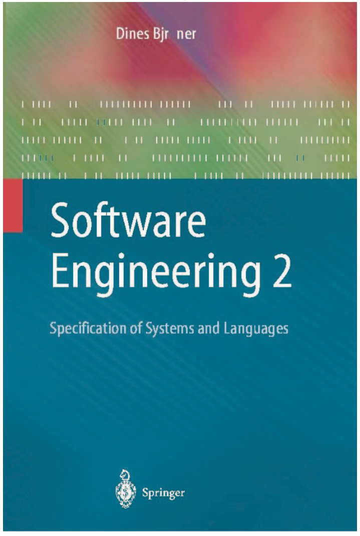
9.4 Future Work

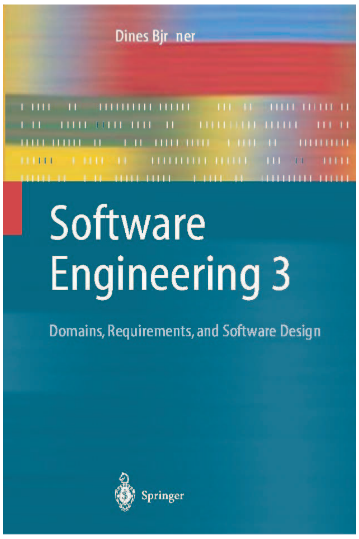
The (i–iv) of the previous paragraph (Sect. 9.3) points out to a serious experimental research activity. It is hoped that work on (i–iv) (Sect. 9.3) will lead to the identification of a number of “lifted” predicates and functions that can vastly simply the masses of the formal predicates, (i), that are now so detailed and specific. Let us indicate what we mean by a first set of “lifted” predicates (we refer to Sect. 3): Chapters 9 and 10 of the ISO Standard appears to address very similar issues: Chap. 9 mostly related to computers and Chap. 10 mostly related to communications. (One could imagine that IBMs representative in the IT System Security effort had focused their contributions wrt. Chap. 9 and that CISCO’s representative in the IT System Security effort had focused their contributions wrt. Chap. 10, respectively.) The technical/scientific (read: engineering/research) questions is therefore: would it be possible and beneficial to “lift” a number of relevant predicates of Chaps. 9 and 10 such that the specific predicates of these two chapters could be simpler expressed in terms of a number of parametrised predicates and auxiliary term functions. These parametrised predicates and auxiliary term functions are what we mean by “lifting”.

Similar R&D issues can be raised wrt. a great number of ‘Code of Practice’ predicates and auxiliary term functions from usually two or more chapters.

In other words: Quite, I think, an exciting PhD topic!







10 What Is Next ?

10.1 A Possible Project Plan

1. Full "formalisation" of all 'code of practice' statements: 6MM
2. Analyse this "formalisation": 6MM
   (a) wrt. all predicates and functions, whether computable or not,
   (b) wrt. an emerging model of the "plant" (etc., Ω), and
   (c) wrt. possible modal operators
   (obligation, permission and dispensation,
   temporality, spatiality, knowledge and belief).
3. Complete a first formalisation of the "plant" (etc., Ω): 4MM
4. Definition of all predicates and functions 6MM
5. Now review all of this in the light of the Gowadia, Farkas and Kudo paper on *Checking Policy Compliance* 3MM
   - Special emphasis is to be put on modalities:
   - Obligation and Dispensation,
   - $\mathcal{E}$tcetera.
6. Rework all models. 6MM
7. All we have then is ***only*** a domain model.

8. Establish a family of requirements for IT Security
   based on these models: 6MM

9. Then: What do you now have?

   - A rather “complete” understanding of IT Security !
   - And a possible widest range of products
     for the support of IT Security

11 What Does a Domain Model Give You ?

1. The models describe all aspects of the real world that are relevant for any good software design in the area. They describe possible places to define the system boundary for any particular project.

2. They make explicit the preconditions about the real world that have to be made in any embedded software design, especially one that is going to be formally proved.

3. They describe the whole range of possible designs for the software, and the whole range of technologies available for its realisation.

4. They provide a framework for a full analysis of requirements, which is wholly independent of the technology of implementation.

5. They enumerate and analyse the decisions that must be taken earlier or later in any design project, and identify those that are independent and those that conflict. Late discovery of feature interactions can be avoided.