Hendrik Heinze
Dr. Khalid Kallow
Harmut Lackner
Dr. Sadegh Sadeghipour
Prof. Dr. Holger Schlingloff
Salko Tahirbegovic
Dr. Hans-Werner Wiesbrock

15

Application and Evaluation in the Healthcare Domain

This chapter deals with the application of the SPES modeling framework to a medical case study: an extended care system (ECS). This system allows patients with serious heart conditions to stay in their home environment while remaining under constant medical surveillance and assistance. The ECS is typical for future telemedical applications, where body sensors and implanted devices work together with an ambient IT infrastructure to guarantee optimal patient-centered care. Design challenges in this case study are the safety and reliability requirements, interface definitions and architecture of the combined system, as well as meeting regulatory demands for life-supporting subsystems while adding further, not necessarily safety-critical components to the system.

15.1 Overview: Application Domain Healthcare

In the healthcare domain, software as an integral part of medical equipment and processes is becoming increasingly important. Software-based systems control and influence more and more medical activities, from first aid to rehabilitation. Similar to the other domains in this book, embedded software plays a central role in the innovation and progress of medical technology. Typical devices are large hospital and laboratory apparatuses, surgery appliances for diagnosis and treatment, and patient equipment usually for the mass market. In all these products, more and more of the innovative functions are realized by software.

Current trends in medical systems

The following are some current trends that are determining and are determined by the development of embedded systems in the field (see, e.g., [Glesner et al. 2007] and [Zauner and Schrempf 2009]):

- Interoperability and interconnectedness: As embedded medical systems obtain more and more communication interfaces, new integrated services are becoming possible. Examples are diagnostic implants that transmit vital data directly to the electronic health record, or the integrated operating theater where the surgeon has central control of various devices and displays.
- Telemedicine and telematics: Embedded devices allow not only remote monitoring of the patient's health, but also the provision of real-time interactions between patient and physician. Robot-assisted intervention devices, such as remotely operable catheters, assist in complex, minimally invasive surgery.
- From diagnosis to therapy: Due to increased sensing and processing power, devices that were formerly only able to observe the patient or provide basic services are now capable of accomplishing complex treatment. One example is a pacemaker with sensors monitoring the heart activity that can now help to cure various cardiopathies.
- Ambient assisted living: Systems designed for the support of elderly or handicapped people in their home can reduce healthcare costs and increase the quality of life. Examples range from simple automated pill dispensers to intelligent wheelchairs and intelligent prostheses that restore physical abilities, such as cochlear implants.

Similar to the aerospace domain, most medical devices are safety-critical in the sense that incorrect software could harm the patient. The more critical the application, the more important the role of the embedded control system within the application and the software used to control it.

In contrast to applications in other industrial domains, such as railway or automotive, the full functionality has to be maintained even in the case of a single fault. It is not an option to shut down the device or go into a "fail-safe state." As a consequence, all functions must be implemented redundant and diverse under the condition of real-time parallel signal processing. Redundancy can protect against random (stochastic) faults during the operation, whereas diversity is used to protect against systematic faults in the design. Fault tolerance is often achieved by parallelism in the hardware. As the system must be shown to be resistant even against common mode errors, the redundant parts must be designed to be largely independent.

Due to the increased complexity of the design, conventional development methods are no longer appropriate for producing code that can satisfy these requirements. As shown in Chapter 9 on modeling quality aspects, model-based design can be used to support the development of such highly critical systems. In the medical domain, development tools also have to respect the above-mentioned special requirements regarding functional safety.

Dramatically increased verification and validation efforts

In parallel to the increase in the complexity of systems, the amount of verification and validation in the development has risen dramatically over the last few years. This is mainly due to the following reasons:

- ❑ The increase in embedded systems technology allows the measuring, influencing, and controlling of more and more physiological parameters of the human body. Therefore, the systems are used in more and more critical applications. This increase in capabilities leads to an increase in complexity in the validation and certification process.
- ❑ The evolution and international harmonization of standards regulates the development processes at a much more finely-grained level than in former times. Therefore, the normative and legal requirements of the development process must be followed and documented to a much higher extent.
- ❑ The increased criticality leads to more and more supervision by authorities. In Europe, for example, Class I medical products can be developed and marketed by the manufacturer in their own responsibility, whereas Class III and IV products are completely supervised by an authority. The manufacturer must provide all necessary evidence and show that he conforms to all essential requirements of the respective European directives.
- ❑ As a social component, the user expectations of electronic equipment are constantly rising, both with respect to their functionality and their reliability. The extremely short product life cycle in consumer

electronics imposes a high innovation pressure also on medical electronic devices, which demands highly efficient confirmation processes.

Corresponding to the increase in verification and validation efforts, the amount of documentation and evidence of compliance that is necessary for approval has risen dramatically in recent years. For example, the main standard of safety in medical products, DIN EN 60601-1 (VDE 0750-1)_1996, had less than 180 pages of requirements in 1996, whereas the most recent version (DIN EN 60601-1 (VDE 0750-1)_2007 with collateral standards) has more than 1800 pages of requirements. US regulations are described in [FDA 2002], for the respective European documents see [IEC 2006] and [ISO 14971].

One of the main benefits of model-based design is that it can help to reduce the manual verification and validation efforts, since it provides a well-founded methodology with a high potential for tool support (cf. [Hungar and Reyzl 2008]). Furthermore, models can be used in the documentation of critical systems and provide a solid basis for assessment by authorities. In the following, we will demonstrate challenges and solutions using a typical example from the healthcare domain.

15.2 Evaluation Case Study: Extended Care System

In this section, we describe a prototypic *Extended Care System* (ECS) that was developed to evaluate the modeling theory of this book. Our prototype of an ECS is typical for a number of similar systems. In particular, our case study exhibits aspects of all above-mentioned current trends in the field (interconnectedness, telematics, therapy, and ambient assisted living). Furthermore, it includes a medical product of the highest safety class. Our system provides an extended range of intensive care for patients with a serious heart condition, such that they can stay in their home environment but remain under intensive care.

15.2.1 Medical Emergency Support Systems

Firstly, we characterize overall requirements and scenarios for medical emergency support systems. In general, this type of system involves three different actors: the patient, the emergency service provider, and the emergency help.

Main purpose of the ECS: quick help in case of an emergency

The main purpose of any emergency system is to provide *quick help in the case of an emergency*. In our case, an emergency can be a fall to the ground, cardiac arrest, apnea, or general immobility. The patient demands a reliable system that, on the one hand detects every emergency; on the other hand, the system must transmit only actual emergencies to the emergency service provider, since unnecessary emergency calls lead to additional efforts and deferral in the handling of actual emergencies.

The task of the emergency service provider is to handle incoming alerts. In order to do so, the data transmitted from the patient side must be comprehensive and understandable so that the provider can evaluate and prioritize the emergency correctly. Furthermore, timely delivery of the data is important to enable prompt reaction. For some systems, the emergency service provider must also be able to constantly monitor all of the patient's relevant vitality data. In medical systems, all personal data of the patient must be treated as confidential and protected against corruption and modification.

In case of an alarm, the emergency service provider may notify and consult emergency help, i.e., a designated doctor or ambulance. The physician has access to the patient's medical record and prescribes further medication or treatment. Additionally, the medical emergency support system may also give advice to the patient for activities to improve his health.

Basic scenario: quick help in the case of an emergency

Our case study is constrained to the scenario *collapse of the patient*. A possible trigger for this scenario is the patient stumbling while walking and subsequently falling down. The patient cannot stand up or crawl on his own and thus remains immobilized. The system detects the collapse when it recognizes a powerful acceleration of the body and limps, a change in the altitude of the upper body (as measured by a barometric sensor), as well as a higher heart rate due to the shock. The sensors attached to the patient's body recognize these changes and exchange their measurement results. Finally, the sensors conclude that the patient has fallen and subsequently alert the emergency service provider.

The service provider receives the emergency alarm and further information about the situation. This information comprises live data from the sensors as well as data from the last minutes before the collapse. The service provider then notifies the emergency help personnel, who take further action.

Negative scenarios

A negative scenario describes what the system must not do. We identified two negative scenarios: the first scenario is that the patient falls and is subsequently immobilized but the system does not detect the collapse (*no alert*). The second scenario is that the patient lets himself sink into his bed, but the system incorrectly detects an emergency and alerts the emergency service provider (*false alert*).

15.2.2 System Structure of the Case Study

The ECS comprises and integrates three components: a ventricular assist device (VAD), a body area network (BAN), and a telematic system (TMS).

System structure of the ECS: VAD, BAN, and TMS

In our case study, the ECS is an extension to an existing heart support system. Thus, we extended an existing VAD by a BAN and TMS. The integrated components form our extended care system for medical emergency support, depicted in Fig. 15-1.

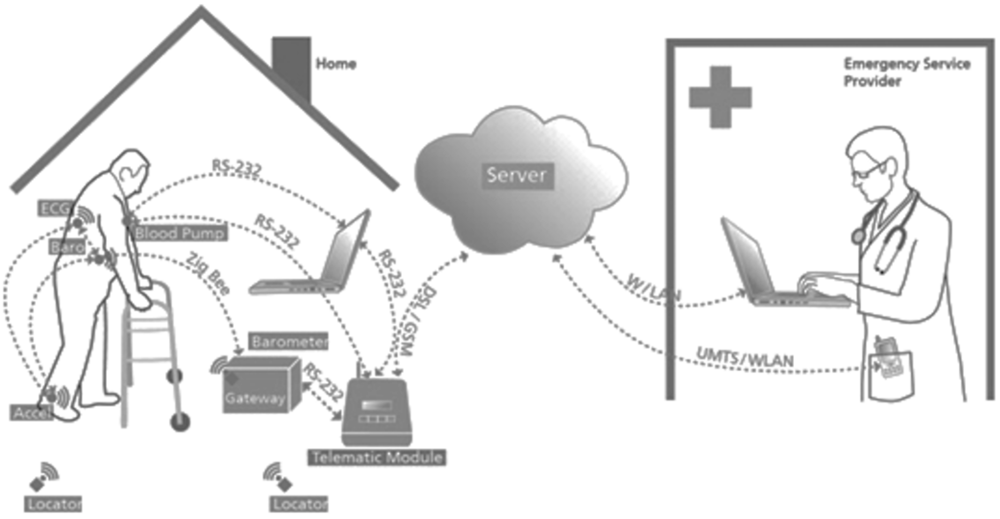

Fig. 15-1 *Extended care system*

Ventricular assist device (VAD)

The VAD supports the patient's heart by augmenting some or all of the heart's pumping capacity. It consists of a blood pump with a microcontroller and a control laptop. The laptop provides a control panel for configuring and monitoring parameters of the microcontroller and displaying patient data and error messages. The control laptop interfaces with the pump's microcontroller by means of a special protocol designed to fulfill the compulsory safety requirements. Cyber-physical modeling of a similar system can be found in [Jiang et al. 2011].

In an ECS environment, the VAD has to be protected against (accidental or malevolent) misuse. Thus, before attaching additional components to the VAD, the following compatibility issues have to be considered:

- ❑ The transfer of data from and to the VAD has to take place via a secure channel.
- ❑ The other components may not at any time change, control, or interrupt the functionality of the VAD.
- ❑ The patient should retain sovereignty over his data; that is, except for emergency situations, a transfer of data may only take place with the patient's approval.

Only if these conditions are met can the BAN and TMS components of the ECS be used as additional components to the VAD. More remarks on high-confidence medical device software can be found in [Lee et al. 2006].

Body area network (BAN)

The BAN consists of various sensor nodes attached to the patient's body and spread around the residence. The sensor nodes form a subnet within the ECS, hence the term *body area network*. A gateway connects the BAN to the TMS and subsequently to the emergency service provider. The BAN has a decentralized structure; there is no central node collecting the data from all sensors as this would form a single point of failure. The nodes may communicate with each other if an exchange of data is necessary to pronounce a more reliable verdict.

The wireless communication among the sensor nodes and the gateway has advantages as well as disadvantages: sent packets may get lost or may be read by anyone within reach. They also have to deal with the fact that other nodes may join or leave the network at any time, expectedly or unexpectedly (e.g., due to power failure). Hence, an authentication algorithm protects the nodes against misuse and a special protocol ensures that if a node is within reach, it will receive its messages eventually.

The BAN component of the ECS is not a medical device in the legal sense, as it is not used for diagnostic purposes in a therapy or medical treatment. Although it tries to identify critical situations and summon professional assistance, it does not actively harm the patient if it fails to do so.

Telematic system (TMS)

For remote monitoring of the VAD and the BAN, the ECS comprises a telematic system. The TMS consists of three components: a telematic module, a server, and the client software. The telematic module transfers the data from the patient side to the emergency service provider. The server hosts the business logic, data backup, and user management. The doctor's client software is for visualization of the patient's data.

The TMS has three main features: online monitoring, alert handling, and data logging. Online monitoring enables the doctor to receive real-time data from the patient side. The live data can be visualized either via a web browser or a Smartphone. Alert handling deals with incoming emergency alerts triggered by the BAN. An emergency dispatcher accepts the alert and decides, on the basis of the data sent, whether further measures have to be taken or the alert is a false positive. Data logging provides the option of transmitting a patient's data to a global data storage. The logged data can be retrieved and evaluated with special software for retrospective diagnosis by a physician.

When integrating the VAD in the ECS, the exception to the third requirement of the VAD, "sovereignty in the transmission of patient data" has to be considered. A patient who falls and possibly loses consciousness cannot and should not intervene in the process of the emergency call. For this reason, the TMS provides a service with which it can independently and autonomously build up a connection to the emergency service provider. The security of this feature must be maintained in the design of the ECS.

15.2.3 Challenges of the Case Study

The case study offers many challenges for modeling, validation, and certification. Since this book describes a specific modeling theory, we subsequently focus on the following aspects:

- ❑ Requirements specification
- ❑ Design and evaluation of system architectures
- ❑ Interface definitions in complex medical systems

Further challenges that are of interest but that are not dealt with in this chapter comprise early validation of the system's requirements, model-based security analysis, testability of requirements, and functional safety for distributed processing in life-sustaining systems.

Safety-critical (Class III) medical products

The VAD is a life-sustaining system and thus is classified as a highly safety-critical product (medical device Class III according to the council directive 93/42/EEC and software category C according to IEC 62304). Therefore, intense validation and verification is needed before regulatory authorities permit release to the market. The BAN and the telematic system are of less concern. One of the challenges of the case study is how to maintain the Class III certificate of the VAD while adding further, not necessarily Class III, components to the environment. It is essential to avoid having to certify the other parts of the ECS as Class III medical devices as well to save efforts for validation and verification. For a successful integration of the BAN and the TMS into the VAD, there must be a guarantee that the combination of the three has no impact on the functionality of the VAD. Therefore, the interfaces between all components have to be designed with great care.

15.3 Example Evaluation Activities in Detail

After describing basic functionality, system structure, and challenges for the ECS in the previous section, we now describe the application of our modeling theory to this case study. To do this, we largely rely on established UML2 or SysML diagrams; see [Raistrick et al. 2004]. In order to manage the complexity of the entire system, different concepts and techniques should be used at various levels of abstraction, as described in Chapters 4 through 7. We describe our modeling using some sample artifacts generated within the development of the ECS.

15.3.1 Requirements Elicitation

The first task in the system development is to identify the boundaries of the system to be developed.

With regard to the ECS, a question relating to the system boundaries was whether calling the emergency help and the subsequent questions of the high availability of and safe communication to the emergency help should be included in the system or be considered as a part of the environment. In the following, we only describe the second alternative.

Modeling the requirements viewpoint with block diagrams, requirements diagrams, and use case diagrams

In accordance with the requirements viewpoint (see Section 4.2.1), we used SysML block diagrams to model the system boundaries (see Fig. 15-2). As described in Section 15.2.1, we distinguish between emergency service provider and emergency help as different actors in the environment. Hence, the emergency call system has to transmit not only alarm messages, but also information on the current patient condition.

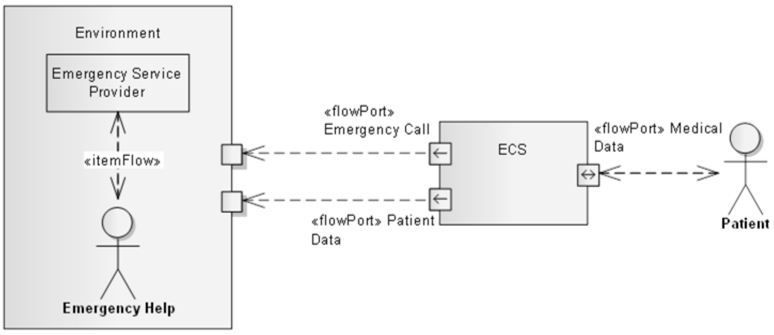

Fig. 15-2 *Context diagram for the extended care system*

According to the requirements process model described in Chapter 4.4, the next step is to define the goals the system has to fulfill. Recognizing the collapse of the patient and sending an alert to the emergency service provider is a main goal of the system. As described above, the emergency service provider and emergency help have limited resources and should be called only if they are really needed. Hence, there are two conflicting goals: calling the emergency service provider immediately in an emergency, and avoiding false alerts. We modeled the goals using SysML requirements diagrams, as suggested in Section 4.2.2 (see Fig. 15-3).

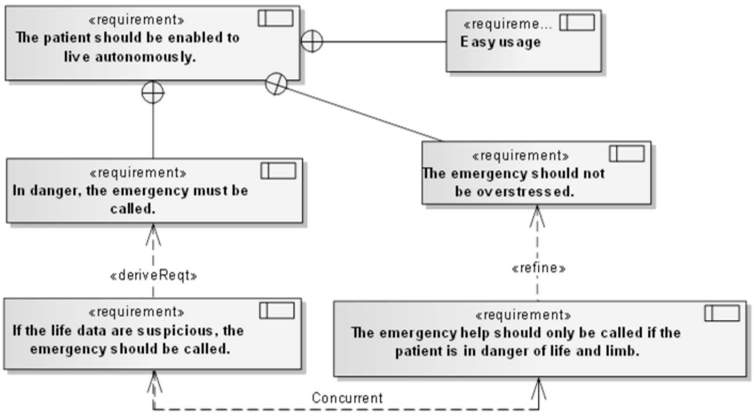
«requirement» The patient should be enabled to live autonomously.
«requireme... Easy usage
«requirement» In danger, the emergency must be called.
«requirement» The emergency should not be overstressed.
«deriveReqt»
«refine»
«requirement» If the life data are suspicious, the emergency should be called.
«requirement» The emergency help should only be called if the patient is in danger of life and limb.
Concurrent

Fig. 15-3 *Concurrent goals for the extended care system modeled by SysML requirements diagrams*

At this point we need a better understanding of the interaction of the system with its environment: How should the ECS behave in a critical situation? What are possible critical use cases? These aspects can be analyzed properly with the help of SysML sequence diagrams and structured by use case diagrams, as suggested in Section 4.2.3. Some use cases are depicted in Fig. 15-4.

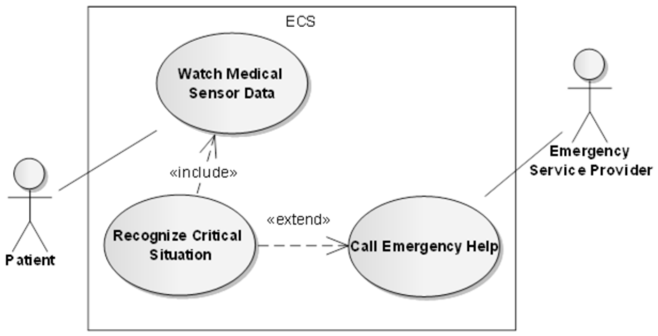
ECS
Watch Medical Sensor Data
«include»
Recognize Critical Situation
«extend»
Call Emergency Help
Patient
Emergency Service Provider

Fig. 15-4 *Use cases of the extended care system*

Sequence diagrams for modeling scenarios

The extended care system should permanently monitor the patient data and notify the emergency service provider if a critical situation occurs. The scenario *collapse of the patient* is demonstrated by the sequence diagram in Fig. 15-5 (the time axis of this diagram points down vertically).

In order to resolve the concurrent goals mentioned above, we decided that a call-back to the patient by the emergency service provider must be performed. If the patient responds to the call-back, the help request generated by the alert can be cancelled by the emergency service provider.

15.3.2 Structural Investigation of the Requirements

The requirements process model, as described in Section 4.4, prescribes that after developing solution-neutral artifacts such as the ones shown in Section 15.3.1, solution-oriented artifacts have to be developed in order to represent the solution concept. After developing the logical structure and behavior of the system, we have to transfer it to a physical realization.

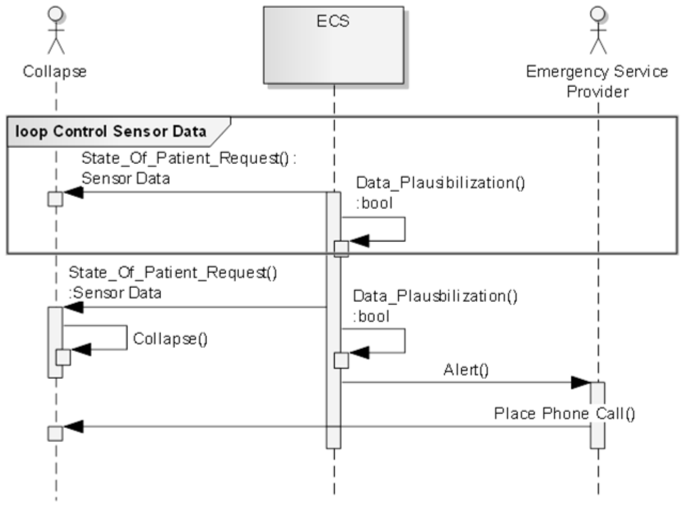
Collapse
ECS
Emergency Service Provider
loop Control Sensor Data
State_Of_Patient_Request() : Sensor Data
Data_Plausibilization() :bool
State_Of_Patient_Request() :Sensor Data
Data_Plausbilization() :bool
Collapse()
Alert()
Place Phone Call()

Fig. 15-5 *Collapse scenario modeled by a sequence diagram*

Modeling patient data by means of class diagrams

As identified in the context diagram in Fig. 15-2, the extended care system must provide all information necessary for immediate help. A system identifier, a unique name for identifying the calling system, and probably other personal data of the patient enable the emergency service provider to look up the patient's anamnesis data in the patient database. Additionally, data on the patient's current state gathered by the extended care system can be important for the emergency help, thus these data should be transmitted as well. To represent these data, we model them with SysML class diagrams, as suggested in Section 4.2.4. The model is shown in Fig. 15-6**.**

15.3.3 Functional Decomposition

Once the requirements have been set, the functional structure of the system as a black box, i.e., from an external point of view, has to be modeled according to Chapter 6.

Deployment diagrams for modeling the functional viewpoint

We used UML deployment diagrams to model the functional architecture of the (BAN part of the) ECS (see Fig. 15-7). These diagrams offer hierarchical structuring and can capture the functional decomposition, similar to the graphical notation suggested in Chapter 6. For this case study, the main function of the ECS is to alert the emergency service provider in case of an emergency. An additional function is to send available diagnostic information. In order to detect a collapse, the system must monitor various physical sensors. In order to send diagnostic data, it must access the data logging storage etc. In this way we described the functional behavior of the ECS in guiding the deployment of the system onto its components.

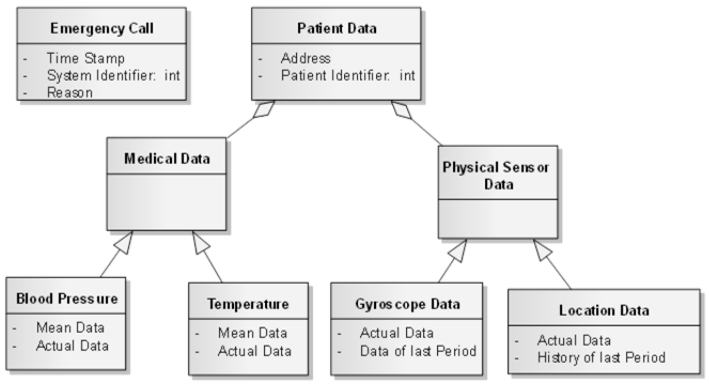

Fig. 15-6 *Information to be sent to the emergency service provider modeled by a class diagram*

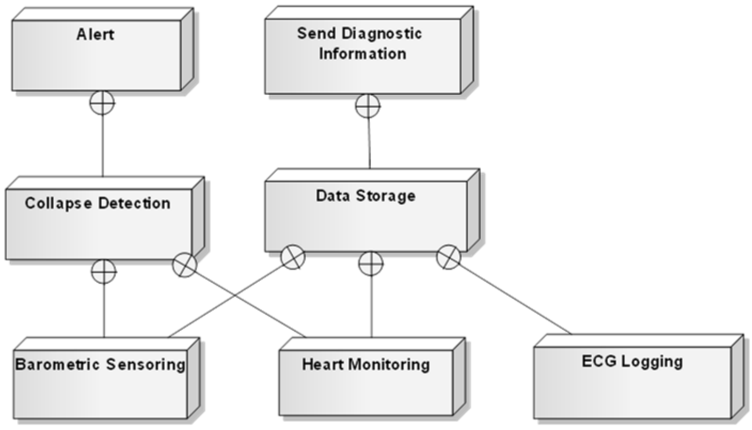

Fig. 15-7 *Functional decomposition of the extended care system using deployment diagrams*

15.3.4 Towards System Design

The previous subsections assumed a black box view of the system to be developed. That is, we looked from outside onto the system. This provides a good understanding of its behavior as expected by the user. In the next step, the implementation is conceived by designing the architecture, based on the logical and technical viewpoints (Chapters 6 and 7). Starting from the functional decomposition of the system, a proper internal structure, as well as a structure for hardware and software resources, is defined using appropriate modeling techniques.

Package diagrams model the technical viewpoint

By considering the BAN from a technical viewpoint, we decided to incorporate mobile and stationary nodes in order to meet the goals "easy usage" and "monitoring of various sensor data." Moreover, the TMS, responsible for the communication with the emergency service provider, is included in the technical structure. It is connected to the sensor nodes via a gateway. In this context, the VAD can be thought of as just another sensor providing information about the cardiac activity of the patient. We modeled the architecture using SysML package diagrams, as shown in Fig. 15-8. The various functions as identified in the functional decomposition model of Fig. 15-7 were then mapped to the nodes that should realize them.

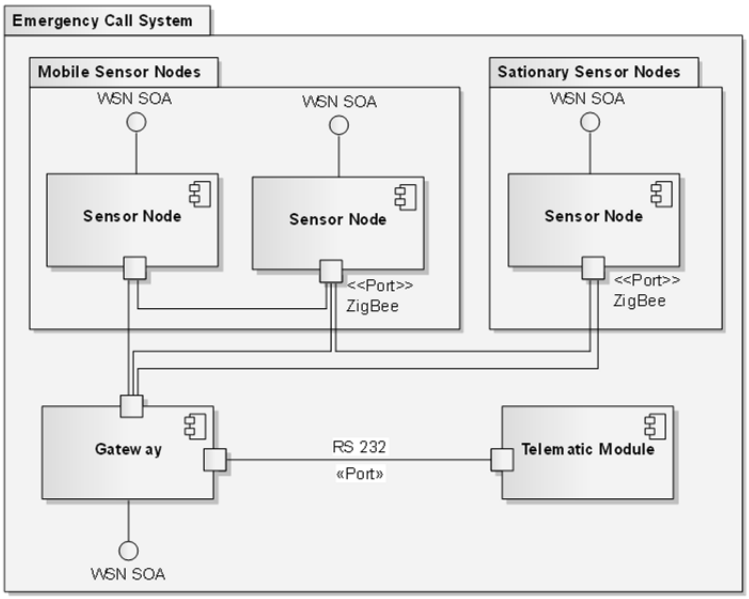

Fig. 15-8 *Architecture of the extended care system using package diagrams*

Integrating legacy code via capsules

For the VAD, most implementation parts already existed. In order to incorporate it into the case study and to enable certification of the complete product, we also designed models for the VAD. Such models are helpful for demonstrating the functional integrity of the system. The general problem is how to integrate legacy code in a model-based development environment. As a solution, we used a reengineering approach. We developed a capsuling concept, where the interfaces between a capsule and its environment were derived from an integrated analysis of code and model. Then, actors were extracted from the tasks in the code. An extended static analysis allowed us to trace the control variables of the actors to derive activity diagrams and state charts. With these generated models, we were able to guarantee that BAN and TMS had no negative effects on the workings of the life-support system.

15.4 Summary

In modeling the various parts of the case studies, we used several modeling notations as defined by the SPES modeling framework. The flexibility of the formalisms supplied was helpful in engineering the models. One challenge turned out to be the still nonexistent tool integration, at syntactic as well as at semantic level. Models that are developed with one tool (e.g., an architecture modeler) cannot necessarily be re-used with other development tools (e.g., test generators), see [Tahirbegovic and Lackner 2011].

Accreditation based on modeling artifacts

Our ECS case study comprised several components, with the patient equipment built on top of an existing VAD system. For such highly safety-critical systems, accreditation by notified bodies is of utmost importance. As mentioned above, the noninterference of the environment with the workings of the VAD must be shown. Here, models and diagrams can be extremely helpful, see also Chapter 9 on modeling safety aspects. Our experiments also showed the importance of modeling a system’s context, as well as unwanted and disallowed use cases. With a large number of models, however, it becomes increasingly difficult to maintain the consistency of the various models and artifacts. In particular, showing that the actual implementation conforms to the different models can only be done partially. Automated procedures such as model checking are helpful only to a certain extent, since many modeling concepts are beyond the scope of a model checker. SysML offers certain constructs that can be used to mitigate this problem. Checking interdependencies between the models still remains challenging.

The increased usage of software in safety-critical medical systems allows a high flexibility for improving and extending a given functionality. In order to be able to use this flexibility and at the same time meet the high safety standards, concepts for modular validation and re-certification are needed. Ideally, validation and verification should start in early development phases, allowing reuse of models, component descriptions, and code fragments together with their respective validation documents in an incremental development process.

15.5 References

[FDA 2002] Food and Drug Administration: General Principles of Software Validation. San Francisco: U.S. Department of Health and Human Services, 2002.

[Glesner et al. 2007] S. Glesner, S. Jähnichen, B. Paech, B. Rumpe, T. Wetter, A. Winter: Strategische Bedeutung des Software Engineering für die Medizin. In: W.-G. Bleek, J. Raasch, H. Züllighofen (Hrsg): Software Engineering 2007, Fachtagung des GI-Fachbereichs Softwaretechnik. Springer LNI P-105, Hamburg, 2007, pp. 25-28.

[Hungar and Reyzl 2008] H. Hungar, E. Reyzl: Software-Entwicklung und Zertifizierung im Umfeld sicherheitskritischer und hochverfügbarer Systeme: Bedeutung modellbasierter und formaler Ansätze für effiziente Entwicklung und Zertifizierung. In: Proceedings of Software Engineering 2008, pp. 291-294.

[IEC 2006] International Electrotechnical Commission: IEC 62304:2006(E): Medical device software – software life cycle processes. First edition, May 2006.

[ISO 14971] ISO International Organization for Standardization: 14971: Medical devices – Application of risk management to medical devices, 2000.

[Jiang et al. 2011] Z. Jiang, M. Pajic, R. Mangharam: Cyber–physical modeling of implantable cardiac medical devices. In: Proceedings of the IEEE, Vol. 100, No. 1, 2011, pp. 122-137.

[Lee et al. 2006] I. Lee, G. J. Pappas, R. Cleaveland, J. Hatcliff, B. H. Krogh, P. Lee, H. Rubin, L. Sha: High-confidence medical device software and systems. In: IEEE Computer, Vol. 39, No. 4, 2006, pp. 33-38.

[Raistrick et al. 2004] C. Raistrick, P. Francis, J. Wright, C. Carter, I. Wilkie: Model driven architecture with executable UML. Cambridge University Press, Cambridge, 2004.

[Tahirbegovic and Lackner 2011] S. Tahirbegovic, H. Lackner: Systematischer Test für vernetzte Medizingeräte: Nicht kapitulieren, sondern automatisieren. http://epaper.konradin-relations.de/medizin+technik/iframe/2011005/. Accessed on: April 3, 2012.

[Zauner and Schrempf 2009] M. Zauner, A. Schrempf: Informatik in der Medizintechnik: Grundlagen, Sichere Software, Computergestützte Systeme. Springer, Vienna, 2009.