The future of goal-based assurance cases

Peter Bishop [1,2], Robin Bloomfield [1,2], Sofia Guerra [1]
[1]Adelard and [2] City University
Drysdale Building, Northampton Square
London EC1V 0HB
pgb@adelard.com, reb@adelard.com, aslg@adelard.com

Abstract

Most regulations and guidelines for critical systems require a documented case that the system will meet its critical requirements, which we call an assurance case. Increasingly, the case is made using a goal-based approach, where claims are made (or goals are set) about the system and arguments and evidence are presented to support those claims. In this paper we describe Adelard's approach to safety cases in particular, and assurance cases more generally, and discuss some possible future directions to improve frameworks for goal-based assurance cases.

1 Introduction

Safety cases are increasingly accepted and mandated as a primary means of communicating the safety requirements, safety management environment and for assuring critical systems. We define a safety case as

> *"A documented body of evidence that provides a convincing and valid argument that a system is adequately safe for a given application in a given environment"* [9]

Although safety cases are generally accepted, there are different ways of constructing such a justification. The three main approaches can be characterised as a "triangle" of:

- Justification via a set of claims about the systems safety behaviour.
- The use of accepted standards and guidelines.
- An investigation of known potential vulnerabilities of the system.

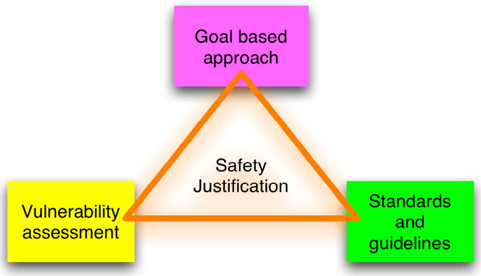

Figure 1: Safety case approaches

The first approach is goal-based—where specific safety goals for the systems are supported by arguments and evidence at progressively more detailed levels. The second approach is based on demonstrating compliance to a known safety standard. The final approach is a vulnerability-based argument where it is demonstrated that potential vulnerabilities within a system do not constitute a problem—this is essentially a "bottom-up" approach as opposed to the "top-down" approach used in goal-based methods. These approaches are not mutually exclusive, and a combination can be used to support a safety justification, especially where the system consists of both off-the-shelf (OTS) components and application-specific elements.

In the past, safety justifications tended to be *implicit* and *standards-based*—compliance to accepted practice was deemed to imply adequate safety. This approach works well in stable environments where best practice was supported by extensive experience, but with fast moving technologies, a more explicit goal-based approach has been advocated, which can accommodate change and alternative strategies to achieve the same goal. This paper will focus on the goal-based approach to safety justification and in particular the approach developed by Adelard. It also discusses possible developments on safety case approaches and how the methodologies and techniques used for developing safety cases can be more generally used in other domains, which leads to the concept of *assurance cases*.

2 Goal-based approaches

Over the past 10 years there has been a trend towards an explicit *goal-based* approach to safety justification. The approach is to support how sophisticated engineering arguments are actually made. This is based on earlier work on argumentation structures by Toulmin [8]. Toulmin's scheme addresses all types of reasoning whether scientific, legal, aesthetic, colloquial or management. The general shape of arguments consists of grounds, claims, warrants and backing:

- Claims, as the name suggests, are assertions put forward for general acceptance.
- The justification for the claim is based on some grounds, the "*specific* facts about a *precise* situation that clarify and make good the claim".

- Next the basis of the reasoning from the grounds (the facts) to the claim is articulated. He coins the term warrant for this. These are "statements indicating the *general ways of arguing* being applied in a particular case and *implicitly relied on* and whose *trustworthiness* is well established".
- Next we may question the basis for the warrant and here Toulmin introduces the notion of backing for the warrant. Backing might be the validation for the scientific and engineering laws used.

The work of Toulmin is the basis of the Adelard goal-based justification approach ASCAD [9], together with our experience of developing and assessing real safety cases particularly early assessment in the 1980s for Sizewell B Primary Protection System and major hazard areas [3]. In the claims-argument-evidence structure used in ASCAD:

- Claims: these are the same as Toulmin's claims.
- Evidence: is the same as Toulmin's grounds.
- Argument: is a combination of Toulmin's warrant and backing.

This relationship is illustrated in Figure 2.

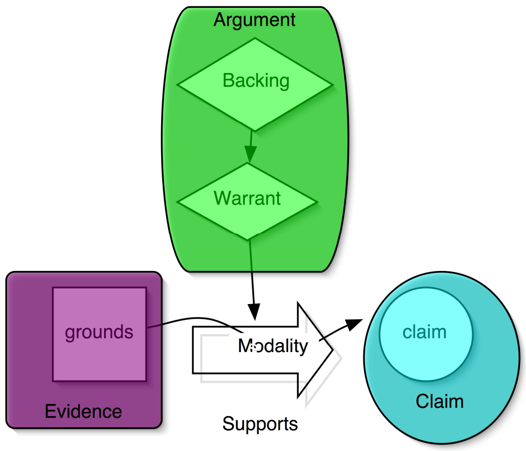

Figure 2: Relationship between Toulmin's scheme and ASCAD Claims-argument-evidence approach

A similar relationship can be demonstrated between the Goal Structuring Notation—GSN [7] and the original Toulmin concepts, where a GSN "goal" is equivalent to claim, which is "solved" by strategies, sub-goals and solutions (which can be related to warrants and grounds). There are also extensions, such a "context" relationships (e.g. to related models).

Tools to support such notations have been developed and are essential if a graphical approach is used. One early development was the SAM safety argument manager [6]. SAM could present and edit safety justifications graphically using the GSN notation, and it also contained supporting tools (like fault tree analysis). The more recent ASCE tool [10] has a more open structure, supporting different notations, like ASCAD claims-argument-evidence and Goal Structuring Notation (GSN) and allows direct links to external documents and tools.

3 Adelard approach to safety cases

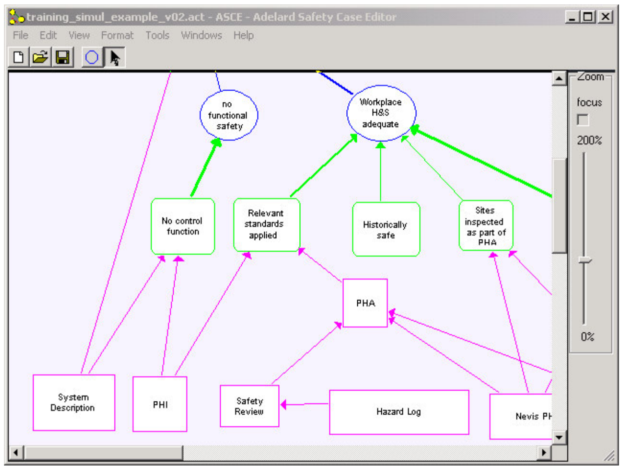
training_simul_example_v02.act - ASCE - Adelard Safety Case Editor
File Edit View Format Tools Windows Help
Zoom
focus
200%
0%
no functional safety
Workplace H&S adequate
No control function
Relevant standards applied
Historically safe
Sites inspected as part of PHA
PHA
System Description
PHI
Safety Review
Hazard Log
Nevis PH

Figure 3: Claim-argument-evidence safety-case

In this section we outline our approach to safety case construction, this is not simply a matter of a suitable notation—the primary concern is the content of the safety case. A general introduction to safety cases is provided by the pages we wrote for the UK IEE Professional Network [1]. Recall our definition of safety case: "a documented body of evidence that provides a convincing and valid argument that a system is adequately safe for a given application in a given environment" [9].

In the Adelard approach, the elements of a safety case are:

- Claims about a property of the system or some subsystem.
- Evidence that used as the basis of the trust argument. This can be either facts (e.g. based on established scientific principles and prior research), assumptions, or sub-claims, derived from a lower-level sub-argument.
- Argument linking the evidence to the claim.

We distinguish different types of argument:

- Deterministic or analytical application of predetermined rules to derive a true/false claim (given some initial assumptions), e.g. formal proof (compliance to specification, safety property), execution time analysis, exhaustive test, single fault criterion.
- Probabilistic quantitative statistical reasoning, to establish a numerical level, e.g. MTTF, MTTR, reliability testing.
- Qualitative compliance with rules that have an indirect link the desired attributes, e.g. compliance with QMS and safety standards, staff skills and experience.

Although it is difficult to make valid sweeping generalisations, we prefer (all things being equal) deterministic arguments to probabilistic to qualitative ones. However the key idea here is that separating arguments is very important.

There are a number of important concepts in deploying the safety case approach:

1. The concept of the top event or the loss event on the environment boundary
2. Definition of system and sub-system boundaries
3. The use of initiating events
4. The risk reduction measures via protection systems
5. The use of functional diversity within the architecture
6. The (often modest) reliability requirements on protection functions
7. Implementation via configured system (COTS)
8. The use of Software Criticality Analysis and criticality and integrity levels

In addition we see that convincing people of safety is essentially a socio-technical process where the case has to be valid and convincing to a range of stakeholders so that issues of communication and consensus building are very important.

3.1 *Safety case structure*

A goal-based approach to justify safety is often a top down method where the claim "the system is safe" is elaborated into subclaims until evidence is available to satisfy the sub-claim. The decomposition is supported by a normally explicit use of arguments to justify the inference being made.

In a real project there are a large number of claims that can be made about a system, its design, development process, operational and organisational context. Each clause in a standard, each component and interface can generate claims. It is a challenge to find a way of analysing and structuring these claims so that the safety justification can be subject to a rigorous review process.

In general we find that safety justifications have two types of claims:

- Claims about the system, e.g. safety requirements, implementation and evidence that the requirements are met.
- Claims about the safety case itself, e.g. claims about the quality of the evidence and the adequacy of the argument–one might call these meta-claims.

These claims are sometimes combined. within a single justification. For example, in CAA SW01 [17] claims about the quality of the evidence appear in the system claims. However in this paper we propose that these should be separated into two claim trees i.e.:

1. A tree where the top-level claim is that the system behaviour is safe.
2. A tree concerning the quality of the safety case.

3.1.1 *Claims about the system*

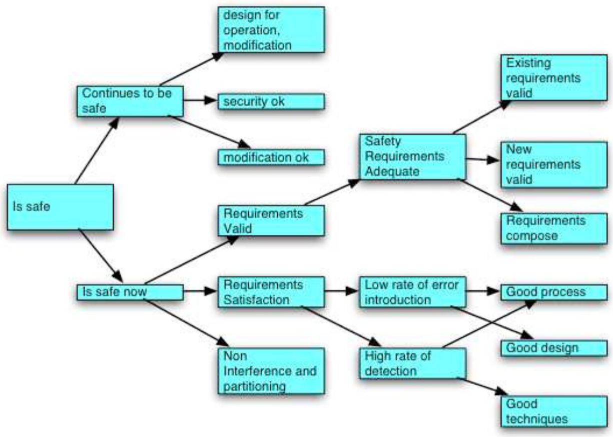

Figure 4: Example of safety-claim structure

Based on a variety of different sources, we can construct a typical tree such as that in Figure 4. The claim decomposition will have to be justified and clearly an argument established that the fault avoidance/detection claims compose to satisfying the safety properties.

Safety properties are derived from a hazard directed approach. System safety analysis identifies hazards; these are amalgamated and abstracted into safety properties. The safety properties can be functions (e.g. shut down of the plant when the temperature exceeds 500 degrees), invariants (e.g. the minimum separation of aircraft is always more than 2 miles) or purely descriptive (e.g. competency and culture). In protection system applications, some of the functions will be "handed down" from plant level analysis and as such can be treated as the starting point for elaborating safety properties. Nevertheless there are still issues in ensuring that the abstraction and models implicit in these requirements match those of the protection system. In our and related approaches [9][17] each attribute of the safety system is considered to ensure completeness:

- correctness
- timeliness
- accuracy
- reliability
- availability
- robustness
- fail safety
- usability
- security
- maintainability
- modifiability

The properties are only safety relevant in a given application context. For example "timeliness" might not be safety relevant for an advisory system, but "accuracy" could be. By focusing on desired behaviour, the argument and evidence are primarily related to the product rather than the process. Typically the evidence for the product comes from:

- analysis (of the system, hardware or software) e.g. static analysis or review
- conventional testing (of components or the overall system, typically to check properties such as accuracy and timeliness)
- reliability testing, such as may be achieved through a statistical approach
- field experience (e.g. analysis field problem reports to identify residual faults or to estimate reliability)

For example, accuracy might be justified by an analysis of the accuracy of the inputs, outputs and the computational algorithm, or by black-box tests using known results. Process aspects (such as standards compliance) can help to provide such evidence (e.g. the results of functional tests) and give confidence that the test results are valid.

3.1.2 *Safety case quality*

The quality of the safety case evidence, our confidence that we are arguing about the right components and have the environment adequately characterised are all important to our overall judgement that the we can deploy a system. Based on the definition in [3] and the goal of configuration consistency in [17] we can develop a tree such as that in Figure 5.

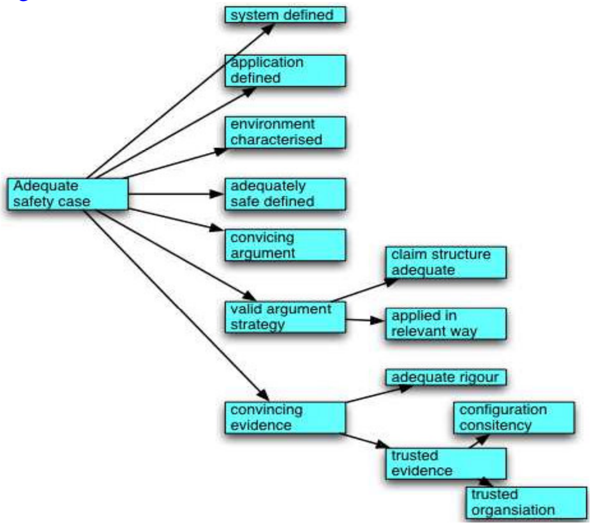

Figure 5: Example of safety case quality claims

Note the importance of justifying the claim structure. Unless the claim structure for the safety system is agreed, there is the risk of debate and contention as the project develops. Also with software safety cases, we often encounter cases where the link between evidence and claim is tenuous and not rigorous enough. An explicit examination of the claim and argument structure will help to reveal this.

3.2 *Assurance of system components*

Common feature of safety case that we encounter is that the system architecture often includes off the shelf (OTS) components. We are undertaking research in this area at the moment to a suitable approach for OTS justifications that can be integrated into an overall safety system justification.

Clearly, it is not possible to say whether a component is safe or not in isolation—safety depends on the context in which the component operates. So we cannot claim that a component is “safe” in any direct sense, but we should be able to justify that the component “does what it says on the tin”. This is a common concept for hardware based systems, which undergo “type approval”, and once approved, are deemed to provide some guaranteed level of service.

Such “type approval” ideas could in principle be applied to software-based and software-only OTS, and could be justified using the same claim-argument-evidence approach.

We have been investigating this approach for the justification of smart sensor, where the claims are about *safety-related properties* of the OTS product rather than the safety of the system as a whole, e.g.:

- correctness (relative to its published specification)
- accuracy
- timeliness
- reliability
- failure modes

We also need to take account of its use within the system as a whole, so we also need to demonstrate:

- non-interference with other components of the system
- suitability of the component for the required application

Typically any OTS component running in the same fault containment region (e.g. the same processor) could interfere with the functions of any other component. So even if the actual function of the component has no direct impact on safe operation of the system, there is potential for interference with other safety-related functions.

Adelard have been involved in assessments of safety-related OTS software to check for the non-interference property. This involved:

- Identification of the safety criticality of the OTS components [4]. This involves identification of the potential interference paths such as shared data or resources.
- Checking for interference between low-and high-criticality components [5] which we term “integrity static analysis” where the potential interference paths are analysed in detail.

More generally, we have produced advice for the UK Health and Safety executive (HSE) on the use of Software of Uncertain Pedigree (SOUP) [16]. For all such components we advocate that the potential hazards of using a component are identified as part of the overall system hazard analysis, and suitable mitigations are implemented (e.g. the use of “wrappers” to minimise the potential for interference or to check the function of the OTS component). These design defences can form part of the justification of the overall system as part of the evidence in justifying the integrity of the whole system.

4 Assurance cases: future directions

While there has been considerable progress over the past ten years in developing a more structured and convincing safety case approach, there are a number issues that still need to be addressed, including:

- the appropriate level of formality
- the relationship to standards
- relevance to non-safety domains

Potential directions are discussed in the sections below.

4.1 Formality and models

Generally speaking, the safety cases we assess (and the ones we construct for clients) are fairly informal, both in the language used and the claims made. Is there scope for greater formality in the construction of a safety case? It would be instructive to consider the models of the system context that the claim refers to because:

- It is easier to reason about claims and evidence within the same context—the models can support valid reasoning.
- The models can make explicit difficult bridges and interfaces in the argument—for example going from quality of the development process to reliability of the product; going from static analysis compliance analysis to reasoning about resource usage and timing.
- The models also assist in structuring sub-claim-trees and help identify the required responsibilities and competencies.

For example, in a recently completed research project [2] we advocate the use of models at the following levels:

- plant /safety system interface
- safety systems architecture
- safety system components
- the operational environment

Claims at any level can only be expressed in terms of entities visible at that level (i.e. in terms of the particular model entities) and claims are supported by evidence about the observable behaviour at that level (e.g. by testing). Alternatively a claim can be supported by sub-claims about more detailed models at the lower levels. For example, a top-level claim at the plant/safety system interface would be expressed in entities visible at the sensor/actuator interface, e.g. if any two sensed temperature values exceed the trip limit value, the output signal will be set to "trip". This might be justified by test evidence at the plant interface, or by analysis of the functions of the components within the system architecture (e.g. that an analogue input has a certain accuracy or a certain failure state). In this case, sub-claims at lower levels of the design are combined e.g. tests of individual components, or formal proof of functionality.

Such an approach might encourage better structured justifications and a clearer identification of the required evidence to support the claim [21].

4.2 Standards

Despite the differences in detail, goal-based approaches are now being adopted in standards. We have had the opportunity of incorporating some of the ideas discussed here in several standards. The UK Civil Aviation Authority software safety assurance standard [17] identifies a standard set of top-level goals for a software based systems which are generic (e.g. specification is valid, specification is correctly implemented, etc.).

We have also contributed to several standards of the UK Ministry of Defence (MOD) [12][13][14], which are linked to MOD guidance on the safety justification documentation [11]. This justification focuses on hazards and their control, i.e. identification of hazards, risk assessment and hazard mitigation, together with compliance to regulations and long term support. These could be viewed as elements of an argument to show that the system will be adequately safe. The MOD also provides guidance on software safety-cases in [11], which again, following our ideas, recommends safety cases structured on claims and evidence, but focused on demonstrating particular specific safety properties. We took the ideas discussed in this paper further in the software part of the new version of Def Stan 00-56 [13] by requiring goal-based safety justification and explicit safety arguments to support the safety claims made.

In the guidance we produced to the UK Health and Safety Executive [16] and that developed in the SHIP project [3], the justification is directed towards the demonstration of *safety properties*, as described above.

Although several standards have adopted goal-based approaches to safety justification, there is still a strong line that advocates more prescriptive approaches, and the educational work needed to support a change of culture remains a challenge.

4.3 Other domains

The type of argumentation presented in this paper is not specific to safety alone, but it could be used to justify the adequacy of systems in different critical applications, including security critical, business critical or service critical. The general case could be called assurance cases, or, if consider more than one attribute and as we have called them elsewhere [19], trust cases or dependability cases.

We have also developed the concept of software reliability case, which is now a MOD [15] and NATO requirement. A software reliability case is defined as “a readable overview of the evidence that the software meets its reliability requirements”. The structure of reliability cases is similar to that described above, where claims about the reliability of the software are made, supported by arguments and evidence.

Similarly, dependability cases are taken to mean the evidence and arguments that are used to support wider dependability claims about systems. Dependability cases enable the user to trust a system, i.e. to have confidence in the service it delivers. Dependability is a generic concept that covers not only safety but also other attributes such as security, availability, reliability and integrity. Dependability cases can be deployed in different application areas, especially for human-computer advisory systems and their development could bring the argumentation techniques described in this paper to even more application areas. They are the subject a several research projects (e.g. DIRC [18]) and were considered in the dependability road-map commissioned by the European Commission [20].

We are also researching security cases for intrusion detection systems. More broadly the concept has applicability to providing justification for Operational Risk assessments within the Basel 2 framework.

5 Conclusions

In this paper we described our ideas on goal-based safety justifications. The discussion considers how to structure a safety case by separating claims about the system from claims about safety case. This structure distinguishes the quality of the system itself from the quality of the evidence and the argumentation that supports the case for safety.

The structure of the safety justifications can also be shaped by the modular assurance of system components, in particular of off-the-shelf components. Rather than claiming that such components are safe, assurance of these components would claim predictable behaviour consistent with the component specification. In addition, non-interference of the components would need to be justified. This component approach has been used in an industrial off-the-shelf product and is being further developed in the context of smart sensors.

Further work is finally discussed, where we consider the use of formality and models to support the validation of the safety case, the relationship to standards and how the safety case goal-based approach can be deployed in other application areas.

6 References

[1] Adelard, “Safety Cases for PES”, http://www.adelard.co.uk/iee_pn/index.htm

[2] Cemsis project. http://www.cemsis.org.

[3] P G Bishop and R E Bloomfield, “A Methodology for Safety Case Development”, Safety-Critical Systems Symposium (SSS '98), Birmingham, UK, Feb 1998.

[4] P.G. Bishop, R.E. Bloomfield, T.P. Clement, A.S.L. Guerra, “Software Criticality Analysis of COTS/SOUP”, SAFECOMP 2002, pp. 198-211, 10-13 September Catania, Italy, 2002

[5] P.G.Bishop, R.E. Bloomfield, T.P. Clement, A.S.L. Guerra, and C.C.M Jones, “Integrity Static Analysis of COTS/SOUP”, SAFECOMP 2003, pp. 63-76, 21-25 Sep, Edinburgh, UK, 2003

[6] J.A. McDermid, “Support for safety cases and safety argument using SAM”, Reliability Engineering and Safety Systems, Vol. 43, No. 2, 111-127, 1994.

[7] Kelly, T & McDermid, J, Safety Case Construction and Reuse using Patterns, Proc 16th Conf on Computer Safety, Reliability and Security (Safecomp ‘97) 1997

[8] S. E. Toulmin “The Uses of Argument” Cambridge University Press, 1958.

[9] R E Bloomfield, P G Bishop, C C M Jones, P K D Froome, ASCAD—Adelard Safety Case Development Manual, Adelard 1998, ISBN 0 9533771 0 5.

[10] Luke Emmet & George Cleland, Graphical Notations, Narratives and Persuasion: a Pliant Systems Approach to Hypertext Tool Design, in *Proceedings of ACM Hypertext 2002 (HT’02), June 11-15, 2002, College Park, Maryland, USA.*

[11] JSP 454, “Procedures for Land Systems Equipment Safety Assurance”, Issue 2, January 2000.

[12] Def Stan 00-55, “The Procurement of Safety Related Software in Defence Equipment” - Parts 1 & 2, UK Ministry of Defence, Defence Standard 00-55/Issue2, August 1997.

[13] Def Stan 00-56, “Safety Management Requirements for Defence Systems”, UK Ministry of Defence, Defence Standard 00-56/Issue 2, December 1996.

[14] Def Stan 00-58, “Hazop studies on Systems Containing Programmable Electronics”. Part 1: Requirements. Part 2: General Application Guidance. UK Ministry of Defence, Interim Defence Standard 00-58, 1996.

[15] Def Stan 00-42, “Reliability and Maintainability Assurance Guides”. Part 2: Software. UK Ministry of Defence, Defence Standard 00-42/Issue 1, December 1997.

[16] P.G. Bishop, R.E. Bloomfield and P.K.D. Froome, “Justifying the use of software of uncertain pedigree (SOUP) in safety-related applications”, Health and Safety Executive Contract Research Report, CRR 336/2001, ISBN 0 7176 2010 7, HSE, May 2001.

[17] CAA CAP 670, “SW01 - Regulatory Objectives for Software Safety Assurance”, CAP 670 Air Traffic Services Safety Requirements. CAA Safety Regulation Group, 1998.

[18] DIRC Interdisciplinary Research Collaboration in Dependability http://www.dirc.org.uk/

[19] “The Pursuit of Trust: Finding a Way Through the High Assurance Landscape” Robin Bloomfield, Keynote at the 27th Annual International Computer Software and Applications Conference, COMPSAC 2003.

[20] Accompanying Measure in System Dependability FP5.8 KAII Roadmapping project June 2002 - May 2003http://www.am-sg.org

[21] RE Bloomfield and B Littlewood, “Multi-legged Arguments: The impact of Diversity upon Confidence in Dependability Arguments”, DSN 2003, pp. 25-34, IEEE Computer Society, ISBN 0-7695-1952-0, 2003.