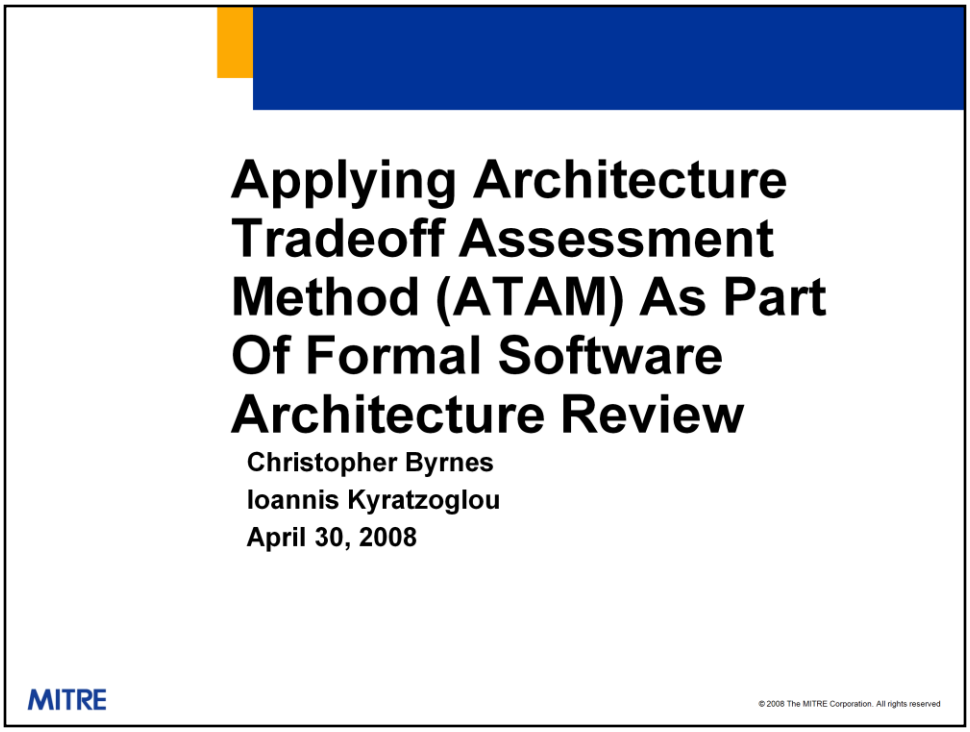
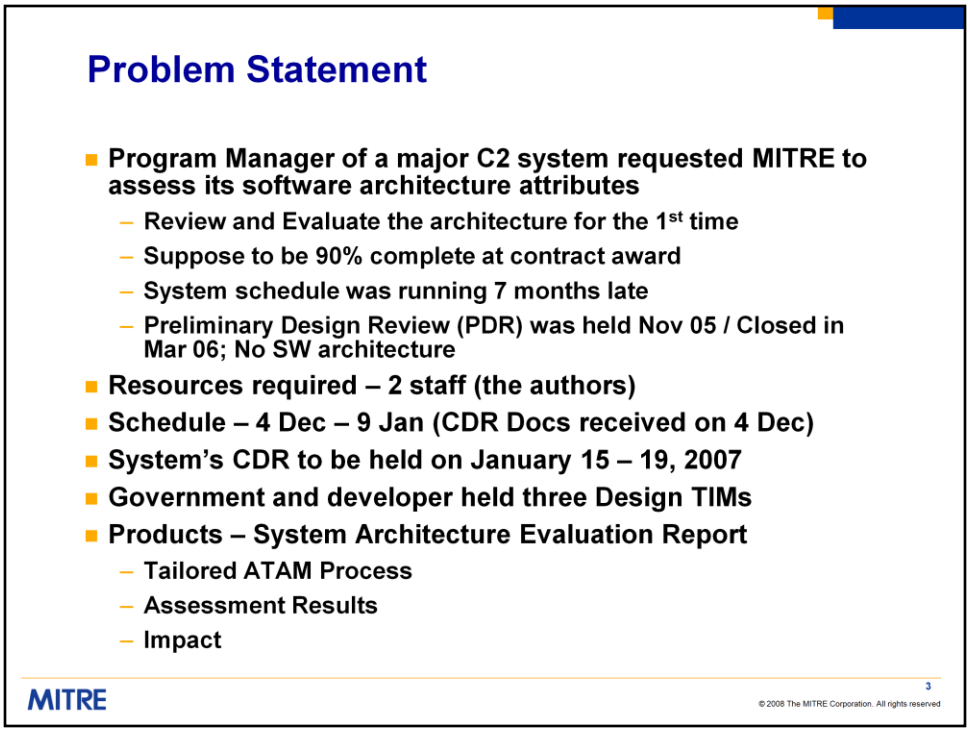


For presentation at the Fourth Software Engineering Institute (SEI) Software Architecture Technology User Network (SATURN) Workshop. The authors can be reached at cb@Mitre.org or ioannis @Mitre.org.

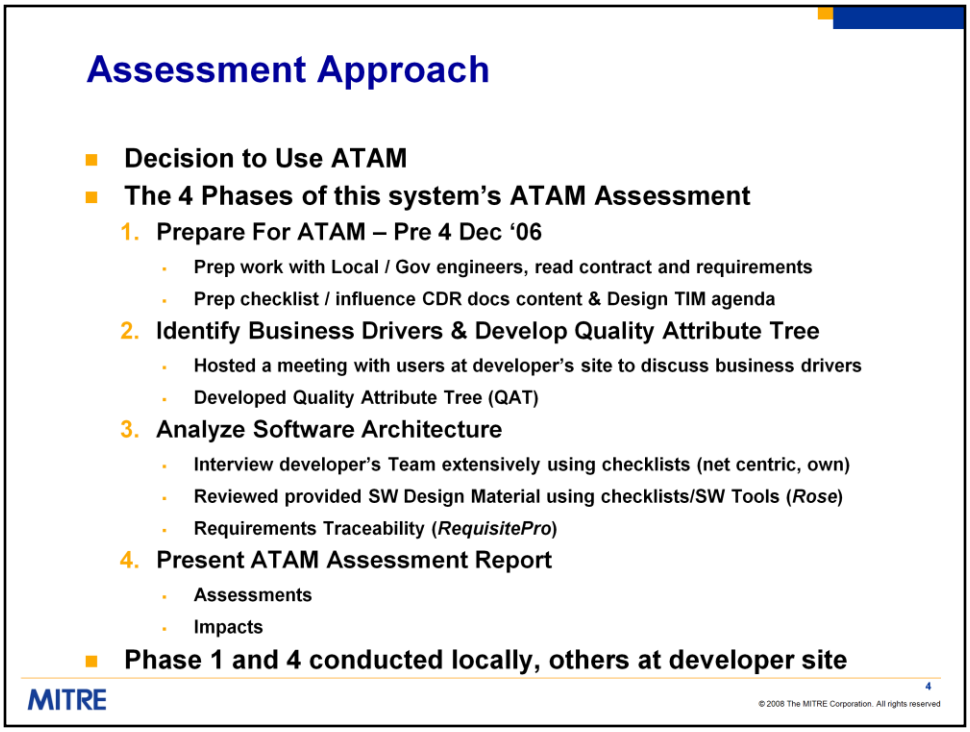
In preparation for a customer's Software System Critical Design Review (CDR); we concluded that an assessment approach based on a hybrid version of the SEI's Architecture Trade-Off Analysis Method (ATAM) would be a good approach for an assessment of this software architecture. This paper will provide ideas on how to apply the SEI's ATAM method within the context of a formal software CDR of a large scale complex software system.

Outline

- CDR Preparations
- SWA Assessment Approach
- ATAM Steps
- Business Drivers
- SWA Assessment
- Available Software Design Products
- Application of ATAM Questions
- ATAM Quality Attribute Evaluation
- Risks in SWA
- Impacts
- Conclusions



MITRE and Government support engineers were requested to assess the software architecture for a customer's project in preparation for a Critical Design Review (CDR). The CDR was a key milestone event, where the contractor had to present and demonstrate evidence that their selected software architecture and detailed design will meet the program's key performance parameters and form the foundation for future Increments. The focus of this software assessment was to investigate at how well the software design met a number of architecture quality attributes such as, configurability, extensibility, scalability, modularity, reliability and interoperability. Particular attention was focused on the interoperability and extensibility of the system since it is intended to be enhanced significantly in the future. We considered several SEI-developed architecture assessment approaches, and concluded that an assessment approach based on a hybrid version of the SEI's ATAM would be optimum for this assignment based on our experience in applying ATAM to other projects. The time and resources available for the assessment during the CDR were limited, so our hybrid approach maximized the use of the available assessment resources and software architecture documentation being prepared for this CDR. There were fewer iterative phases in this hybrid approach as there is in the full ATAM, this allowed an architecture assessment with requiring as much interaction with the customers.



The ATAM defines four major phases numbered 0 – 3. The activities associated with each phase were tailored to address the CDR needs. The activities are described in greater detail in the subparagraphs below. Briefly, an ATAM Phase 0 consists of an assessment team overview presentation of the proposed software architecture approach and presentation of the initial set of questions. Phase 0 laid the groundwork for the ATAM's Phase 1 and Phase 2, leading to a software architecture assessment report produced during Phase 3. In the subsections below, each Phase is described in more detail, followed by a description of how each phase was applied to this project.

During Phase 0, MITRE and Government read the required contract documents (e.g., Statement of Work), the associated requirement documents (Capabilities Description Document, Technical Requirements Document) and the integrated master schedule. The team extracted the related paragraphs that identity the architecture qualities and the types of products that will be presented by the contractor during the CDR. The first item will ensure that we are working within the legal bounds of the contract and the latter will provides us an idea of the products and the architecture presentation style. The next step was to work with the program manager to influence the contents of the CDR material. In parallel, we were preparing the assessment checklists. Using these assessment checklists as our guide, we were able to propose tailored CDR documents and a CDR agenda that will fit the SW Assessment checklist framework.

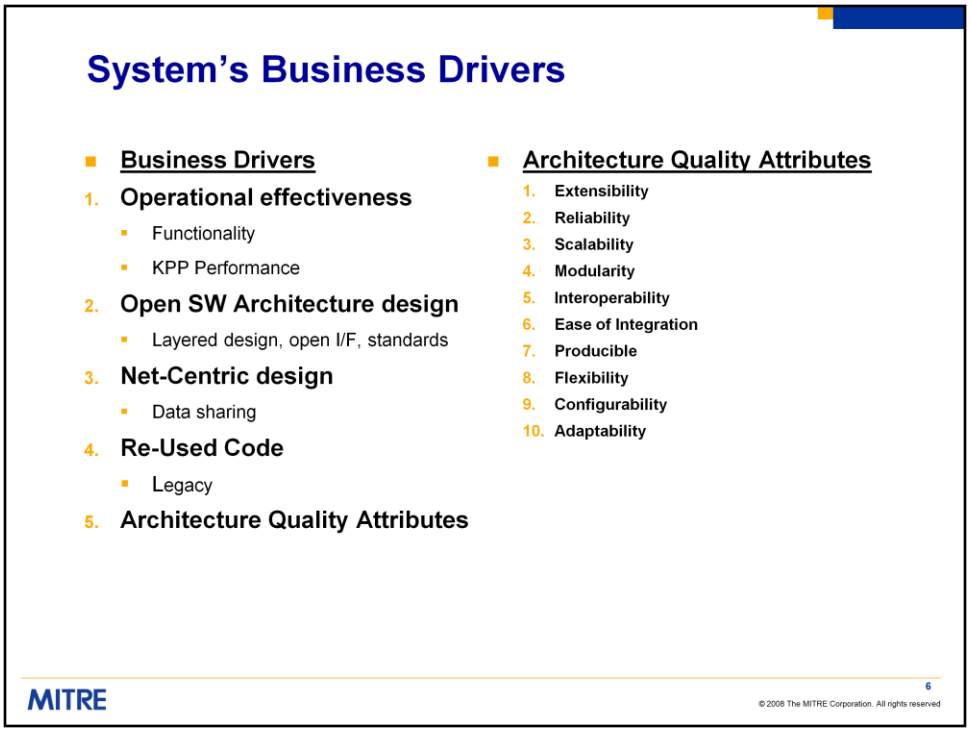
ATAM Steps

Step	Action
1	Present ATAM methodology to the Program manager, contractor and users; completed at a project Design Technical Exchange Meeting (TEM) prior to this CDR.
2	Identify Project's Business Drivers and design decisions. The business drivers identified the reasons we need this capability and the key mission drivers that will also drive design decisions. Done at Design TEM.
3	Present key functions and mission flows that will lead to lower level functional decomposition. Present top-level software architecture details on how software components are layered as part of a software architecture, software interfaces and software interactions . Also done at Design TEM.
4	Identify software architecture principles and approach, as provided by existing design products or Unified Modeling Language (UML) schematics. Also done at Design TEM.
5	Generate ATAM Software Architecture Quality Attribute Tree, based on written requirements and interactions with users. This is what would be looked at in more detail prior in Phase 2 of the ATAM.
6	Analyze Software Architecture Approaches (as identified in step #3); where the information provided in steps #3 to #5 would be reviewed.
7	Provide Software Architecture Modification Scenarios, where some of the UML scenarios and the anticipate architecture modification scenarios would be applied to this software architecture.
8	Analyze Software architecture capability to evolve based on future Increment requirements and under an Engineering Change Proposal Scenario Note in this hybrid approach, this ATAM step was skipped.
9	Present ATAM Results both to the project's managers and during the CDR.

MITRE

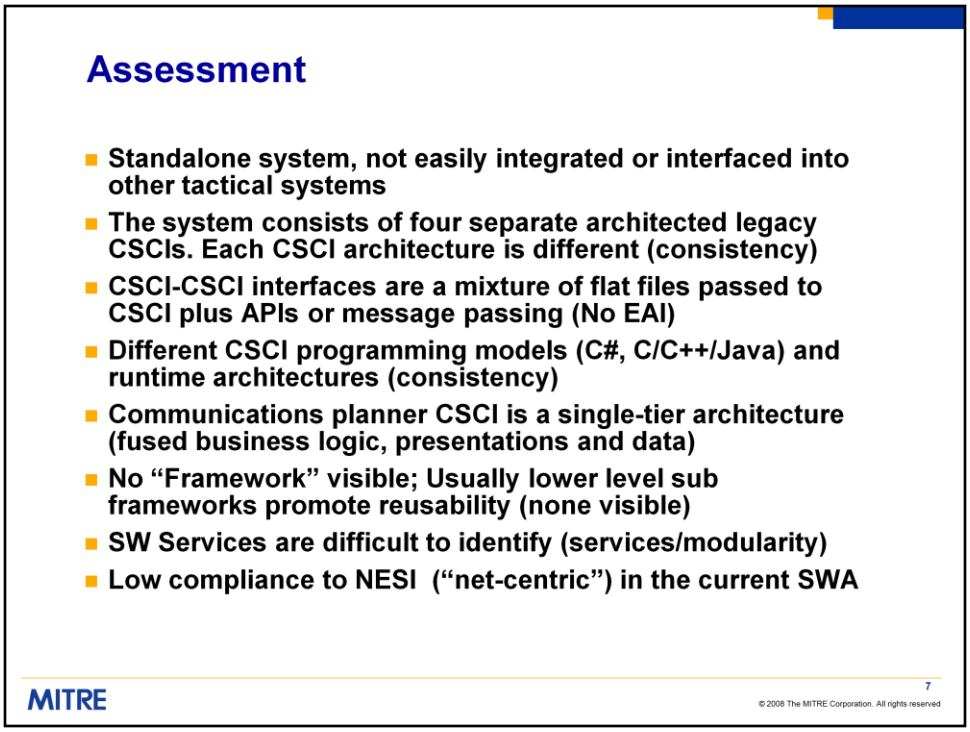
Phase 1 covered development of ATAM "business drivers" (which the application domain stakeholders and customer believe are important) and the identification of software architecture approaches. The hybrid approach to ATAM would mean mostly simplifying the software architecture and presentations. We would still go through the same 9 ATAM steps, but with less formality than what is described in the SEI's ATAM reference. Other ATAM reports that MITRE has participated in during the past have shown the ATAM to be a very "heavyweight" approach; the assessment of this project by necessity of the resource limitations and schedule demands had to be more of a "lightweight" assessment. The 9 ATAM steps followed in this assessment are shown below in this slide.

Typical checklists included a standard ATAM questionnaire; software quality assessment; net-centric checklist for NESI compliance, data management; information assurance, Internet Protocol version 6 (IPv6); DoD Architecture Framework (DoDAF) architecture questionnaire; software best practices; programming models; software framework. The lists served as a backbone for further exploration and questioning. We were also able to get a feel from the users what are the most important mission capabilities and most important architecture quality attributes matched against them.

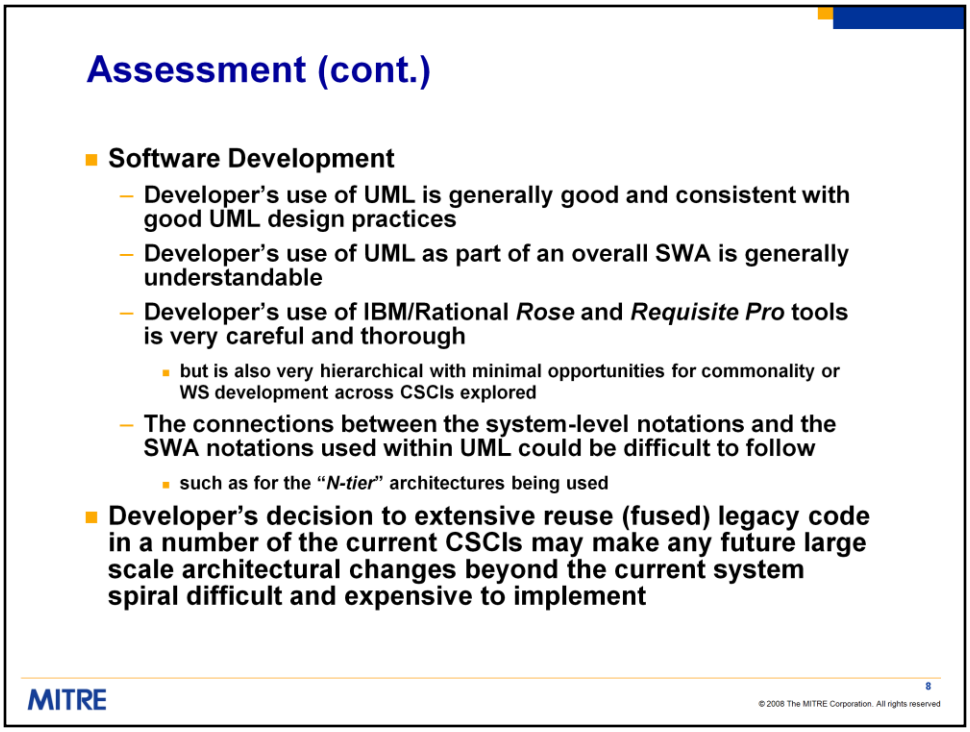


In Phase 2 we analyzed the various software architecture products, particularly the architecture usage and modification "scenarios" that developer's use of UML should be producing. One such candidate change scenario, to assess software architecture extensibility, was the Dynamic Interface Reconfiguration Capability. This new capability, introduced in the next software increment, will allow to dynamically change interfaces and its parameters without an orderly system shutting down. The net-centric compliance scenario was used to assess the interoperability attribute of the project's software architecture. During the TEM, the ATAM team talked with the developers, system users and other stakeholders to gain concurrence of the scenario(s) we would use in this ATAM Phase 2 to assess the robustness of the software architecture.

During the ATAM team's meeting with these stakeholders, we were able to conduct Phases 0 and 1 of the ATAM, covering steps #1 - #6 in the ATAM list shown above. The ATAM "business drivers", identified in step #2 of the previous slide, were established by the system users as "exit criteria" for the CDR and come directly from the system's Statement of Work (SOW). The table here provides a list of the project's key quality attributes.



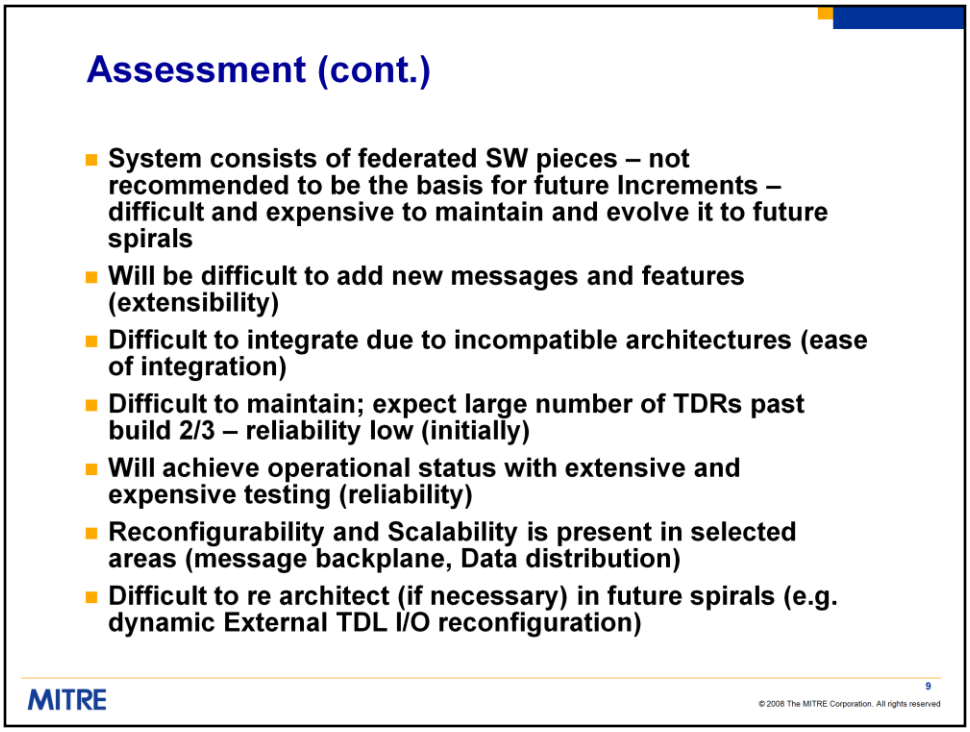
Phase 3 of the assessment was to interview the stakeholders and engineers and assemble and evaluates the data require to generate the ATAM report for the customer. Based on these answers and a review of these software architecture products, the ATAM team arrived at some preliminary conclusions that assessed the contractor's software architecture.



The developer's use of UML is generally good and consistent with good UML design practices. While extensive, it is possible to trace through most of this system spiral's software architecture, and the developer's presentations at the CDR should be understandable to most system stakeholders.

The developer's use of UML as part of an overall software architecture is generally understandable; with nearly all UML diagrams carefully noted and annotated to document assumptions and special cases in the threads of behavior.

The developer's use of IBM/Rational *Rose* and *Requisite Pro* Computer-Aided Software Engineering (CASE) tools is very careful and thorough, but is also very hierarchical with minimal opportunities for commonality or Web Service (WS) development across system components explored.



The connections between the system-level notations (such as for the "*N-tier*" architectures being used) and the software architecture notations used within UML could be difficult to follow. This was particularly true for the DoDAF "views" being developed.

There is very limited "net-centricity" in the current software architecture. Adding the NR-KPP may prove to be difficult and expensive, and this software architecture has limited current support for net-centric notions.

The developer's decision to extensive reuse code in a number of the current components may make any future large scale architectural changes beyond the current spiral difficult and expensive to implement.

Available Software Design Products

Document Description	On Contract	Available
Concept of Operations (CONOPS)	Yes	Yes
Technical Requirements Document (TRD)	Yes	Yes
Software Test Plan (STP)	Yes	No
System Requirements Specification (SRS)	Yes	Yes
Interface Requirements Specification (IRS)	Yes	Yes
Interface Control Document (ICD)	Yes	Yes
Requirements Traceability Matrix (RTM)	Yes	Yes
System Subsystem Design Description (SSDD)	Yes	Yes
Interface Design Document (IDD)	Yes	Yes
Software Design Description (SDD)	Yes	Yes
Software Development Plans (SDP/IMS)	No	No
Capability Maturing Model Integrated (CMMI) Assessment Reports	No	No

MITRE

10

Part of the ATAM preparation work done prior to the first step was to see what available documents could be provided by the developers. This table lists the document artifacts required to conduct the evaluation. The documents should be available in both paper and electronically; with columns on the right side indicating which were on contract and available to the ATAM team.

Available Software Design Products (cont.)

Artifact Description	On Contract	Available
Rational Rose Architecture Design Data (or Data)	No	Yes
Rational *Rose* Reports (e.g., Consistency Reports)	No	Yes
Requisite Pro attribute and other metrics reports	No	Yes

MITRE

11

The ATAM team requested from the contractor to make available the following information shown in the table above in electronic form. The contractors Software engineers were able to generate the standard output from their Rose and Requisite Pro CASE tools as part of the software architecture. However, the ATAM and other stakeholders team at the Design TEM did not have any electronic access, and the contractor have no plans to provide such access by the time of CDR.

Application Of ATAM Questions

#	ATAM Team Questions	ATAM Team's Assessment based on Contractor Response
1	What are the driving architectural constraints, and where are they documented?	These are all fully documented in the Software Design Document (SDD).
2	What component types are defined?	Component types are described in the SDD. But there is not a unifying idea of generic types of component in this SW architecture.
3	What component instances are defined by the architecture?	The software architecture is not Object-Oriented (OO), but there are software architecture diagrams that show where multiple instances of SW modules
4	How do components communicate and synchronize?	Sequence diagrams defined interactions and synchronization between components.
5	What are the system partitions?	Generally along traditional component boundaries.
6	What are the styles of architectural approaches?	A mixture of N-tier software architecture and UML-based class diagrams.
7	What constitutes the system infrastructure?	Lower level tiers define the system infrastructure, but commonality is hard to find with all the legacy modules in use.

MITRE

12

The ATAM Team also prepared a list of questions that the contractor should respond during the ATAM discussions, with final answers due at the CDR. The checklist has 14 major questions.

Application Of ATAM Questions (cont.)

8	What are the system interfaces?	Describes in the IDDs.
9	What is the process/thread model of the architecture?	A few components have a defined thread model, but in (too) many cases a singled threaded model of control and data remains from legacy code.
10	What is the deployment model of the architecture?	Use UML deployment diagrams to describe them.
11	What are the system states and modes?	Higher level components came with standards that defined major modes, which were also further captured in both the HMI mockups and UML use case mini-specification alternative control flows.
12	What variability points are included in the architecture?	For a number of S/W architecture variations there are specific hooks for when new functionality is enabled. Some components have interface design that describes the initialization files. But in other cases, variability would have to be custom coded in the future.
13	How far along is the architecture's development?	Did not get a chance to run Rose "consistency checking" reports that would verify this, but the software architecture looks fairly complete for new services. But legacy code is only partially described.
14	What is the documentation tree?	The documentation tree is fully described.

ATAM Quality Attribute Evaluation

Quality Attribute	Concerns	Description	Importance / Difficulty
Openness	Well defined interfaces for each architecture layer	The system does not have well-defined open interfaces.	High
Flexibility	Accommodate changes in functionality	Based on the two Change Scenarios, the architecture is not flexible. It does not accommodate changes easily and there will be a cost and schedule impacts to include these capabilities into the baseline.	High / High
Scalability	Increase number of interfaces	The Software architecture scales fairly well up to a pre-defined number of simultaneous connections. The system can scale up to 50 users	Medium / Medium
Modularity	Framework Architecture Programming model is composed of independent, open, domain-specific processing modules	The system consists of four separate legacy applications. Each application architecture is different (consistency) The overall architecture consists of federated SW pieces – not recommended to be the basis for future Increments – difficult and expensive to maintain and evolve it to future spirals The software architecture is based on the existing set of components extending or adding any of these with any sort of WS-based modules would be a major change.	High / High

MITRE

14

This table shows an ATAM-based summary of the software architecture quality assessment attributes and concerns. Each of the quality attributes (shown both in the list of Business Drivers on an earlier slide and the first columns above) have had their importance and difficulty supplied during on-site discussions held during the CDR, with the Description of each attribute in this table was supplied by both discussions with the customers and developers during the CDR plus reviews of the available documentation.

ATAM Quality Attribute Evaluation (cont)

Interoperability	Ability to work with other systems or products	The system is interoperable with the external systems identified in the DoDAF System View 6 and uses the specified DoD technical standards. The system does not use the net-centric services but point-to-point interfaces thus making difficult and expensive any interface changes.	High / Medium
Extensible	Accommodate growth and changes in future increments	The Software architecture does not have any sort of performance engineering model for covering the "200% growth" factors in key performance attributes The software and data architecture will not be easily extensible to accommodate changes in the specified scenarios	Medium / High
Consistency	Consistent user display and user interaction models	The Software architecture (with a few variations across different components) tend to use UML consistently. There was also consistent usage of HMI mockups for each major function.	High / Low
Producible	Ability to produce the different product deployment configurations	The system architecture allows the system to be produced and deployed in the three specified configurations.	Medium / Medium

Risks In Software Architecture

- Being spread out over so many files made the software architecture and specific portions within it (such as the RTM) hard to browse and search.
- Lack of anything resembling any UML “activity” or “collaboration” diagram in these SSDDs tended to reinforce the hierarchical description of the use cases and made more abstract behaviors (that spanned multiple use case alternatives) harder to see.
- Too many assumptions about single threaded components and services.
- The data layouts of some of the packets being exchanged could be hard to follow in some of the (legacy) tabular formats used.

Impacts

- Discussed results with System's customer
- Manage Expectations
  - Expect large cost increase per change proposal or Increment change
  - Expect schedule and high cost due to low-quality legacy software - low reliability and high deficiency report rate.
- Proposal
  - Improve reliability (Improve Defect Report Density)
    - Improve Integration process (Sw Integration)
    - Improve Unit Testing
  - Apply Sw Development & Test Quality Metrics

- System did not meet the Open Architecture requirement

Conclusions

- SEI's ATAM was an important method for assessing software architectures against known quality criteria established by the contract and the stakeholders
- ATAM team was able to apply this method with minimal changes to ongoing development and presentation plans
  - Fitting a hybrid version of the ATAM into existing deliverable products and meeting schedules
- Same detailed familiarization approaches called for by the ATAM were also useful as a general preparation step for the CDR
  - Allowed stakeholders to focus on areas of greatest interest to them
- Project received a series of detailed recommended changes to this system architecture to improve its quality attributes