Extensibility as a Collaboration Enabler: A Case Study for Group-Context-Aware Mobile Applications

Marc Novakouski
Grace A. Lewis
Enrique Sánchez

Software Engineering Institute
Research, Technology and Systems Solutions (RTSS) Program
Advanced Mobile Systems (AMS) Initiative
May 9, 2012

Software Engineering Institute | Carnegie Mellon

This material is based upon work supported by the Department of Defense under Contract No. FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

Agenda

Background

Architecture Drivers and Scenarios

Architecture Decisions

Extensibility as a Collaboration Enabler — Results

Conclusions

Agenda

Background 

Architecture Drivers and Scenarios

Architecture Decisions

Extensibility as a Collaboration Enabler — Results

Conclusions

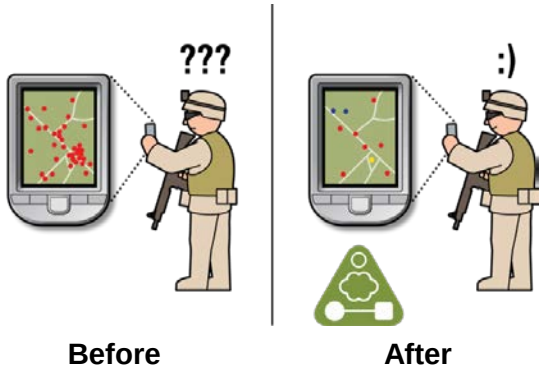
Group-Context-Aware Mobile Applications [1]

Context-aware mobile applications are capable of sensing and responding to changes in their environment or context

Group-context-aware mobile applications integrate the individual's context with that of nearby individuals operating as part of a group or unit, such as in the military or first responder situations

Integrated context is used to enhance the precision of information provided to users as well as a more complete picture of the status of a mission

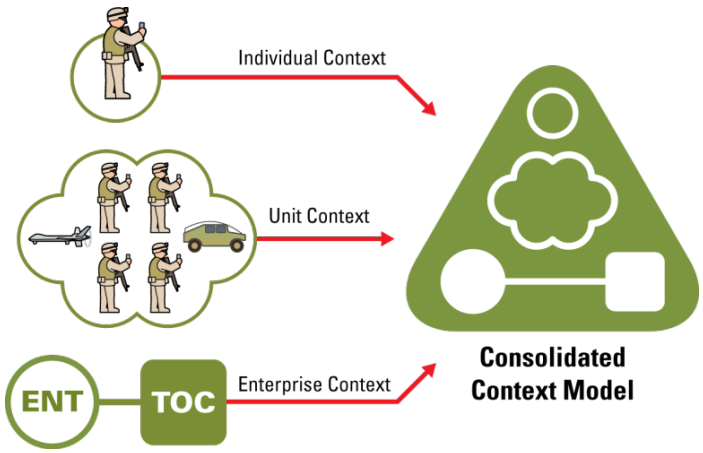
- Goal is to produce a capability that can sense as much of the emerging context as possible and apply that context to filter data such that only the most relevant information is displayed



Group-Context-Aware Mobile Applications 2

Desired characteristics of the solution include

- Capturing context information on a handheld device in a non-intrusive manner
- Extending the sources of contextual information beyond location and time
- Storing context information and disseminating this information to peers
- Capturing and using context information efficiently without imposing an unreasonable burden on handheld device resources
- Integrating local and group context information and only displaying information that is of relevance to the individual and mission according to pre-defined rules

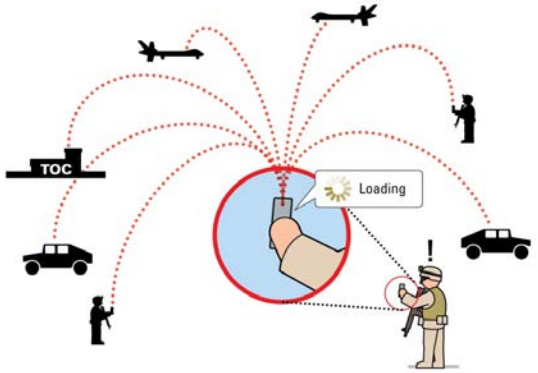


Motivation

One of the more interesting results of this work has been the ability to leverage the architecture to support collaboration

By identifying extensibility scenarios early on in the design process, we were able to construct an architecture that supports multi-organizational collaboration to construct and evaluate different pieces of the architecture

- context data models
- context data storage
- context sensors
- context reasoning engines and rules
- context views



This has allowed us to reach out to researchers from multiple universities and industry, resulting in synergistic research and development, furthering the goals of all participants

Agenda

Background

Architecture Drivers and Scenarios 

Architecture Decisions

Extensibility as a Collaboration Enabler — Results

Conclusions

Business and Architectural Drivers

Business Drivers

- Opportunistic integration of new technology
- Ease of integration with components produced by collaborators
- Applicability of architecture to different edge-enabled applications

To meet business drivers we defined **extensibility** as the main architectural driver.

Extensibility Scenarios

#	Name	Attribute Concern
1	Add a New Sensor	Separation of Concerns
2	Add a New Sensor	Modifiability
3	Add a New Communication Mechanism	Separation of Concerns
4	Add a New Communication Mechanism	Modifiability
5	Add a New Context Event / Action	Separation of Concerns
6	Add a New Context Event / Action	Modifiability
7	Add a New Context View	Separation of Concerns
8	Add a New Context View	Modifiability

Scenario 3: Add a New Communication Mechanism

Scenario	Add a New Communication Mechanism	
Attribute	Extensibility	
Attribute Concern	Separation of Concerns	
Scenario Refinement	**Stimulus**	Developer
	Stimulus Source	Developer identifies a communication mechanism that can be used to share context data with other mobile devices
	Environment	Developer is sufficiently comfortable with application to make changes in a reasonable amount of time
	Artifact	Communications Manager of the context-aware system
	Response	Communications Manager is changed to implement message passing using the new communication mechanism
	Response Measure	Aside from communication-mechanism-specific code, only the Communications Manager is changed to accommodate the new communications mechanism.

Scenario 5: Add a New Context Event / Action

Scenario	Add a New Context Event/Action	
Attribute	Extensibility	
Attribute Concern	Separation of Concerns	
Scenario Refinement	**Stimulus**	Developer
	Stimulus Source	Developer identifies a new event that can be detected by examination of context data
	Environment	Developer is sufficiently comfortable with application to make changes in a reasonable amount of time
	Artifact	Context Engine of the context-aware system
	Response	Context Engine is changed to detect the conditions for the event and generate a new action when it is detected
	Response Measure	Only the Context Engine is changed to allow for detection of events and generation of actions

Agenda

Background

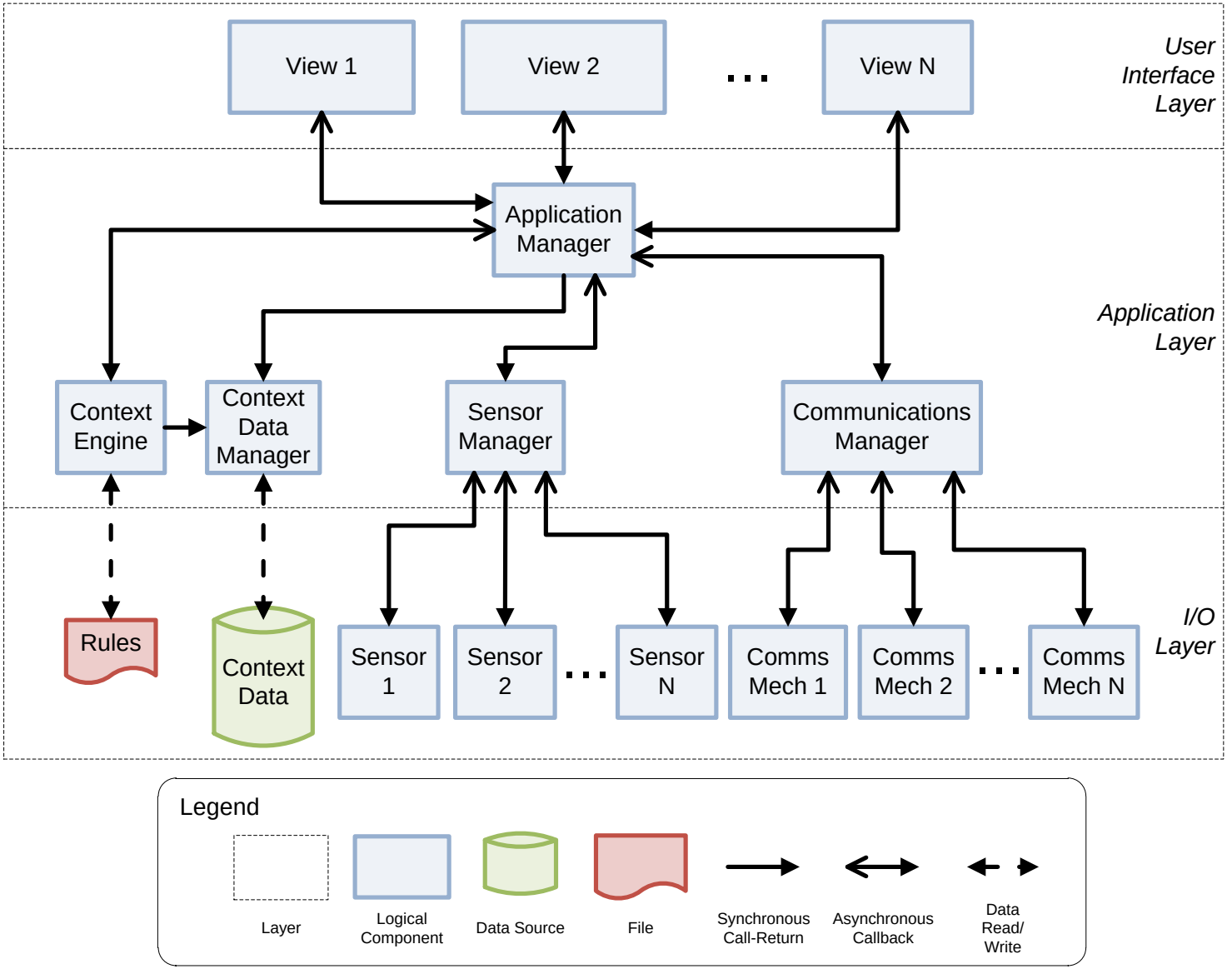
Architecture Drivers and Scenarios

Architecture Decisions 

Extensibility as a Collaboration Enabler — Results

Conclusions

High-Level Reference Architecture



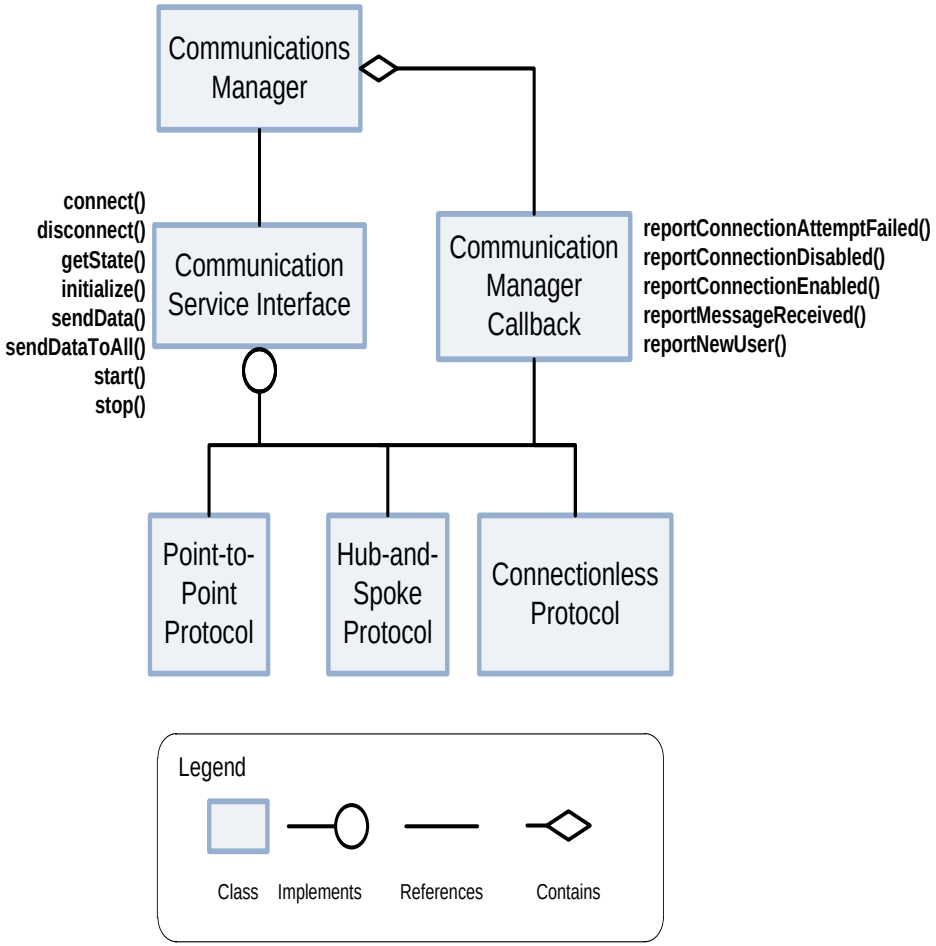
Architectural Decision 1: Communications Interface

Challenge: Integration of very different communication mechanisms

- Different protocols support different use cases
- Target hardware is unknown
- Need to adapt to target network capability

Solution

- Common service interface provides generic communication methods and callbacks that individual protocols can adapt as necessary
- Allows any sequence of communication events to account for differences in protocols



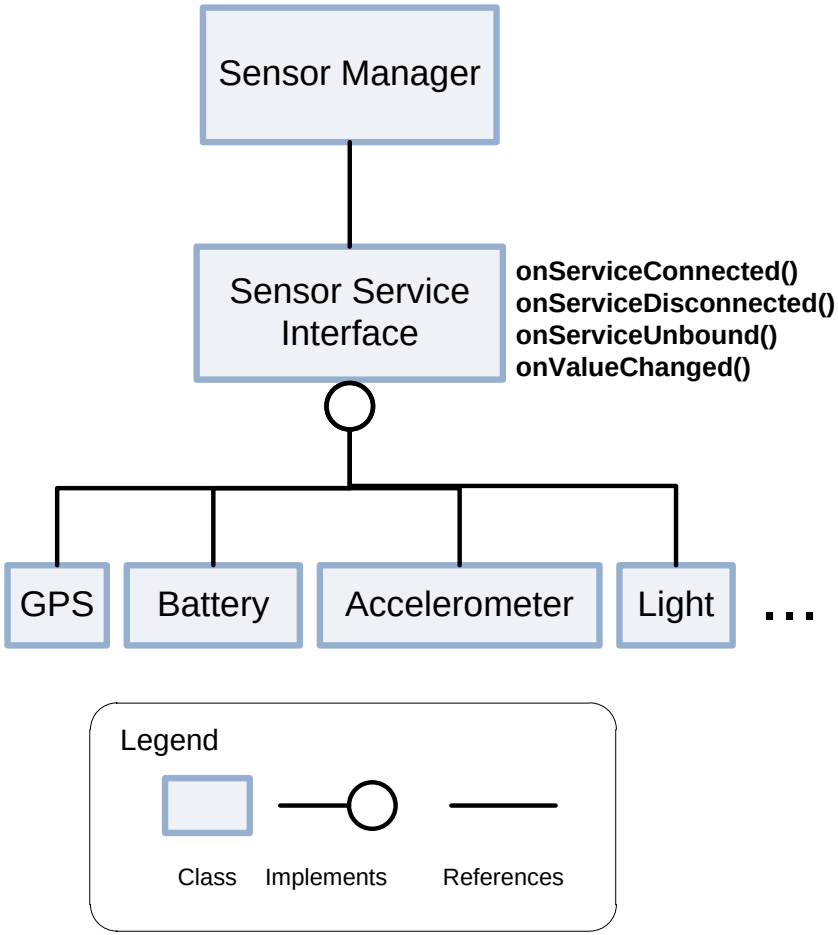
Architectural Decision 2: Sensor Interface

Challenges

- Integration of any current or future available sensor
- Control of sample rate and change threshold

Solution

- Common sensor interface provides generic communication methods that individual sensor implementations can adapt to as appropriate



Problem

Peer review of the architecture raised the issue that a single thread would cause high-frequency sensors to overwhelm the application

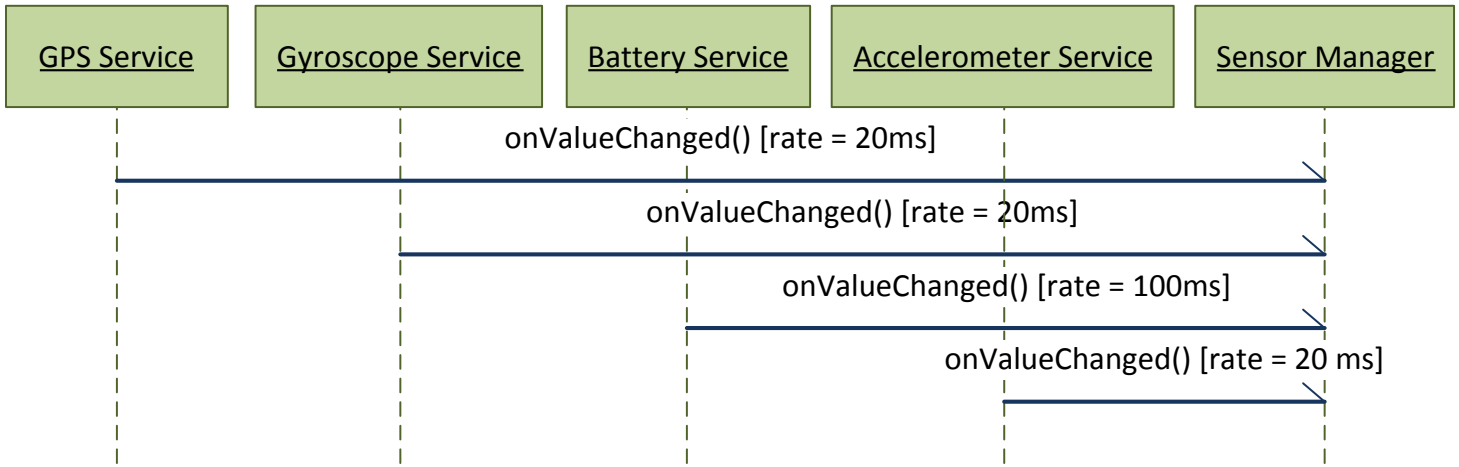
Simple experimentation demonstrated that this was indeed a problem

Solution

- Sensors implemented as Android Services (processes separate from the application)
- Communication via IPC to insulate application from high poll rate impact

Tradeoff

- Higher complexity in sensor implementation although interface hides as much as possible



Architectural Decision 3: Context Model "At the Center"

Challenges

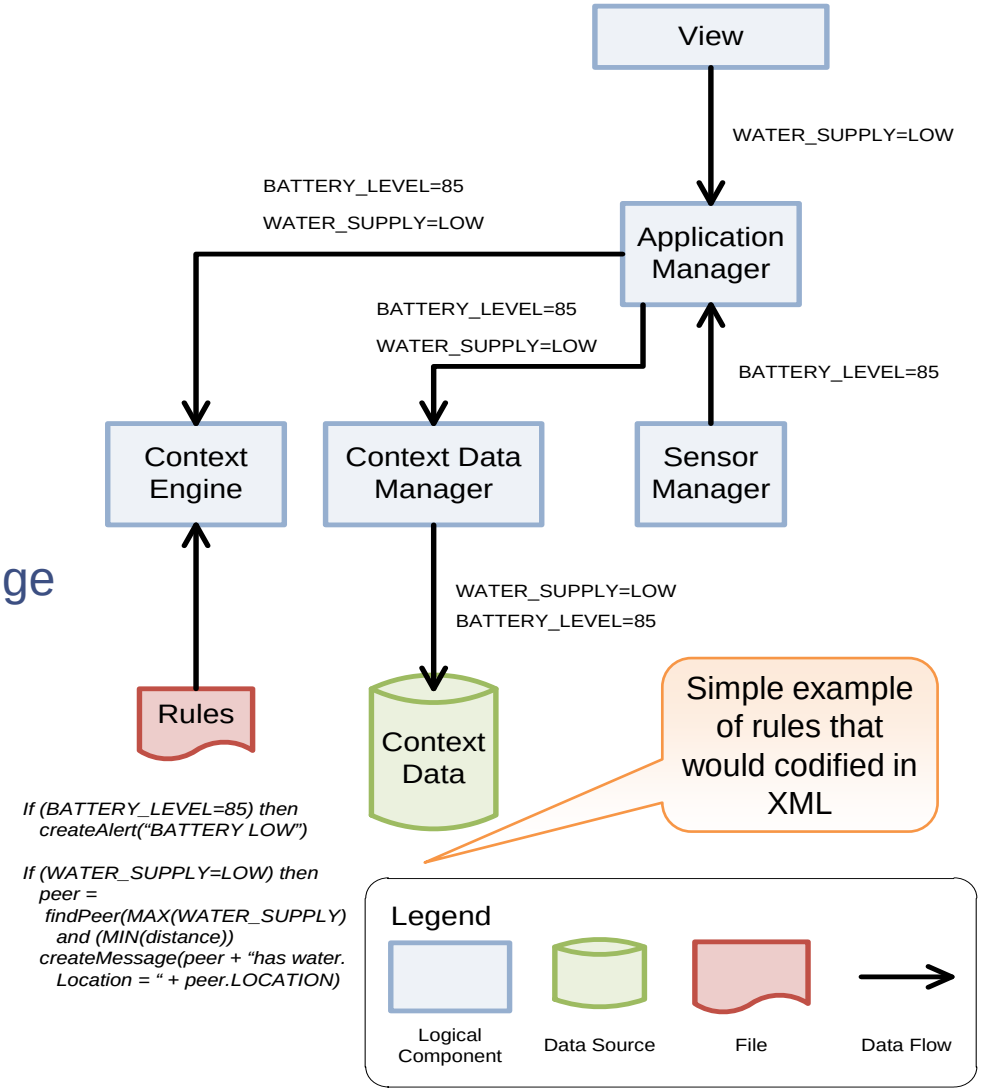
- Easy creation of rules based on contextual data captured via sensors or user input
- Standardized rule processing

Solutions

- Generic and extensible context model that can handle a wide range of situations, environments, data
- Standardized rule set read by application from XML file

Tradeoff

- Both sensors and views have to know the context model element that they are affecting — strong coupling



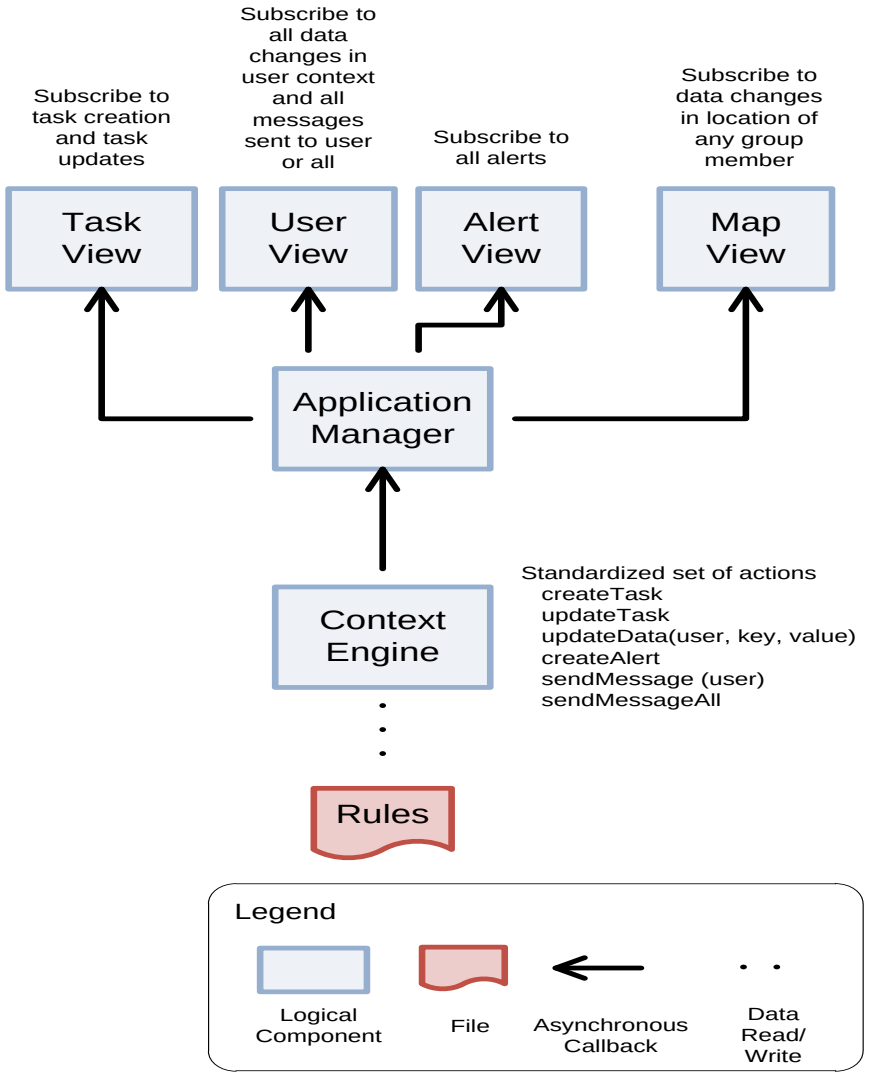
Architectural Decision 4: Standardized Messaging

Challenge

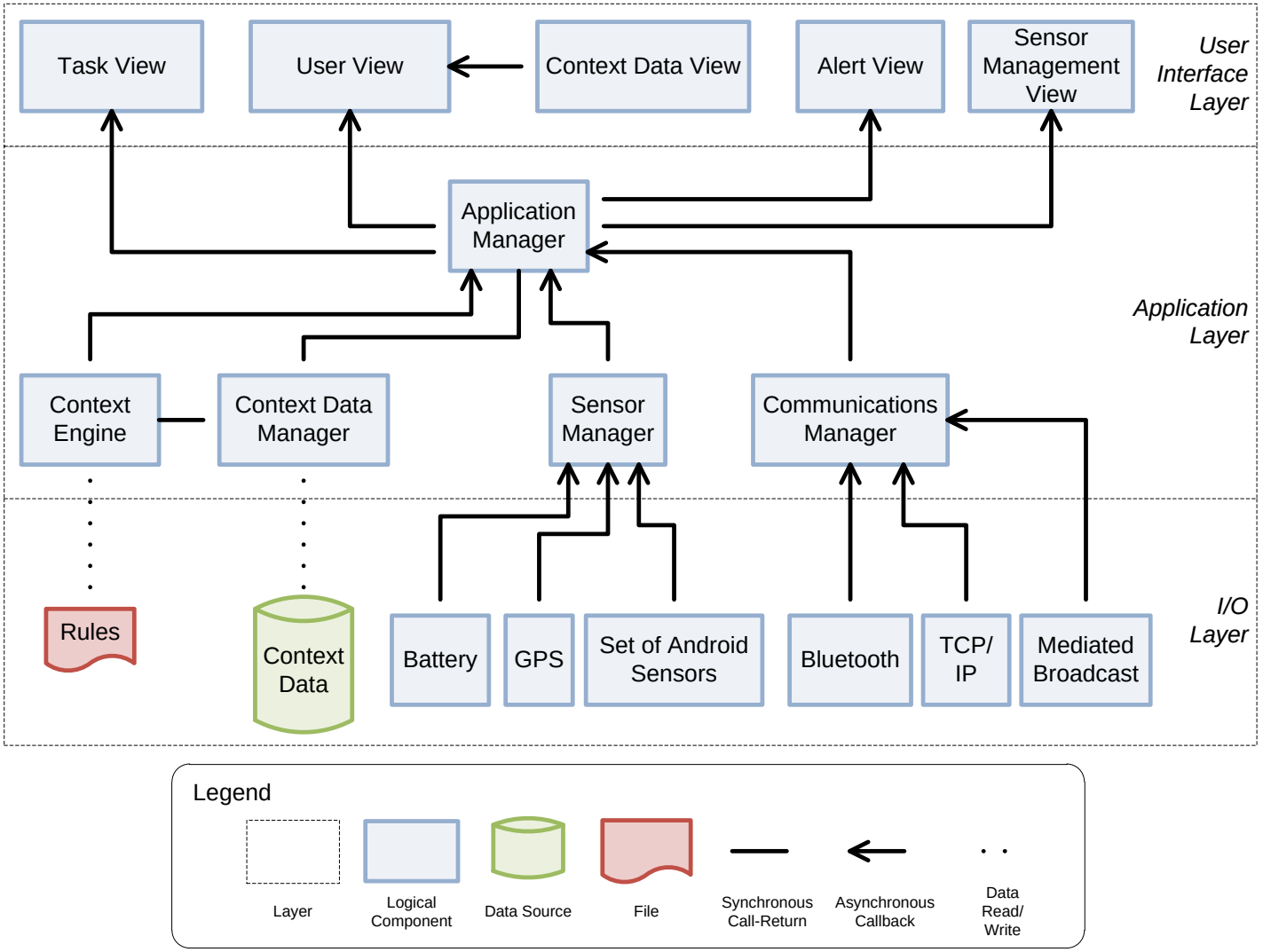
- Easy creation of views that can capture and/or display context data

Solution

- Publish/subscribe interface
  - Standardized set of actions that can be created by the context engine as the result of fired rules
  - Application manager publishes actions created by context engine as standardized events
  - Views subscribe to events



First Responder Application Architecture



Agenda

Background

Architecture Drivers and Scenarios

Architecture Decisions

Extensibility as a Collaboration Enabler — Results 

Conclusions

Results 1

The extensible architecture enables productive collaboration

- Sensor and communication service interface enable 3rd parties to contribute new/novel sensor and protocol implementations
- Standardized rule set approach allows enables adaptation to different context data models
- Standardized messaging enables easy integration of new context data views

Results 2

Collaborators at GMU were able to modify their unique communication protocol to interface with application architecture in just a few weeks

Collaborators are working on developing a group context data model, unconstrained by implementation details and without affecting our progress in the meantime

Collaborators within SEI planning to integrate related projects for QoS management, code offloading, and end-user programming with no foreseen complications

Agenda

Background

Architecture Drivers and Scenarios

Architecture Decisions

Extensibility as a Collaboration Enabler — Results

Conclusions 

Conclusions

Extensibility as an architecture driver enables productive collaborative research and development

Scenario-driven architecture design along with peer architecture evaluation is useful even for small projects

- Concrete definition of quality attribute requirements
- Early identification of risks and tradeoffs

Contact Information

Marc Novakouski

Research, Technology and Systems Solutions (RTSS) Program
Advanced Mobile Systems (AMS) Initiative

Software Engineering Institute
4500 Fifth Avenue
Pittsburgh, PA 15213-2612
USA

Phone: +1 412-268-4274
Email: novakom@sei.cmu.edu
WWW: http://www.sei.cmu.edu/staff/novakom