Internet-driven and Web-supported Software Integration: Cooperation in Distributed Teams

Chris Ireland
The Open University, UK
Panelists

- **Mihaela Iridon**, Candea LLC, USA
  - "Defining implementation processes and testing frameworks, sharing data and service contracts, and team collaboration in general."

- **Manuel Cabral Reis**, University of Trás-os-Montes e Alto Douro/IEETA, Portugal
  - "Web-based systems to help managing primary students’ homework: teachers’ & parents’ cooperation."

- **Chris Ireland**, Open University, UK
  - "Challenges in distributed teams: shared design, partitioning the build, software architecture, configuration management, team management, methodology"
Cooperation Challenges

- Leadership (Central vs Distributed)
- Roles and Responsibilities (Clearly Defined vs Loose)
- Methodology (Approach: Structured vs Fluid)
- Language (Verbal and Design: One vs Many)
- Culture (Similarities and Differences)
- Decision Making (Dictate vs Consensus Building)
- Trust (Reliability?)
Software Integration Challenges

- Economies of Scale: Library of resources, etc.
- Disparate Resources: Resource identification
- Choice of a Resource: Resource selection
- Resource Control: Configuration management
- Resource Lifecycle: Change management
- Trust: Skills and communication
- Priorities: Time and Availability
Discussion Topics

- The use of a "mock up" as a way of identifying integration issues.

- Developing user friendly mobile apps: software development is not just about coding challenges.

- How to prepare graduates for jobs in software integration. Not just focus on programming skills but also implementation skills and industry knowledge.

- The status of programming as a career. Provide career opportunities within software development rather than moving into people management.

- The skills required of an integrator.
Panel on FASSI/AFIN
Internet-driven and Web-supported Software Integration: Cooperation in Distributed Teams

Manuel J. Cabral S. Reis

IEETA/University of Trás-os-Montes e Alto Douro (UTAD)
Escola de Ciências e Tecnologia (ECT)
Departamento de Engenharias

Venice, Italy, August 2015
- Schools as “distributed teams and organizations”.

- Different computer systems (hardware and OS):
  - desktop/laptop;
  - smartphone/iPhone;
  - tablets/iPADs;
  - ...
Different types of apps:

- office (create, edit, share documents);
- sharing (e.g., dropbox);
- social networks (e.g., facebook, twitter);
- organizer & management (e.g., TeacherKit);
- Learning Management Systems (also online);
- Internet (google, (wiki)pedias, ...);
- ...
Web-based systems to help managing primary students’ homework:
- teachers’ cooperation;
- parents’ cooperation;
- teachers/parents communication and interactions.

Really user-friend apps.
SOFTWARE SYSTEMS INTEGRATION

TECHNOLOGY, PROCESSES, AND PEOPLE

Mihaela Iridon, Ph.D.
Cândea LLC (USA)
TECHNOLOGY PERSPECTIVE

• Service-Oriented
  • Microsoft
    • WCF
    • Web API

• Service Contracts: Operation and Data
  • Clean, Stable, Extensible
  • Operation Contract Stability:
    • General-purpose operations: “HandleNotification”
  • Data Contract Stability:
    • Serialized data: HandleNotification(string notification)
Microsoft-Centric Implementation

Benefits

- ASP.NET, MVC 5
- Bootstrap
- Angular
- C#.NET
- EF 4/5, ADO.NET
- Autofac, MEF
- Automapper
- WCF, IIS-hosting
- SQL 2008 R2
- ElasticSearch*

Integration Middleware

Payroll

- C#.NET
- IIS, AppFabric
- Windows Server
  - Service Bus 1.1
- SQL Server 2012
- ADO.NET

HR
INTEGRATION (BOUNDARY) SHAPE

• Symmetric Topology (Client-Client)
  • Broker-based architecture (hub-and-spoke)
  • Bus architecture
  • Integrators are all peer clients in the overall system, communicating via an integration middleware

• Asymmetric Topology (Client-Server)
  • Specialized technology or service providers; also S/PaaS
  • Example: Alcatel’s Genesys – call center technology
    • Hardware abstraction
    • Technology abstraction
    • Service abstraction
TEAMS’ COLLABORATION PERSPECTIVES

• **Case 1:** Enterprise Integration: integrating existing systems
  • One team per integration slice
  • One (small) team for integration middleware
  • Shared access to project epics, stories, tasks, bugs
  • Daily standups*

• **Case 2:** Completely separate application lifecycle for platform provider and integrator entity
  • Interaction with support team (higher tier of support/developers of platform)
Alleviating Dependencies & Maintaining Development Progress

• Unit testing
  • Integration components
  • Integration paths and couplings (joints)
  • Models and behavior

• Simulation of components to integrate with

• Simulation of domain-specific data
  • Example: message/data generators

• Simulation of integration middleware

• Dependency injection to mock both data and behavior

• Code generation/automation