Intelligent Software Engineering: The REAGENT Framework

Using Agent Technology For Better Software Development
Decision Systems Laboratory

Research Team

- Chee Fon Chang
- Dr Aditya Ghose
- Dr Thomas Meyer

Funded by

- The Australian Research Council (Large Grant) $153,000
Software Engineering

Developing software that meet requirements have always been problematic at best.

Some of these problems are:

- Involves multiple stakeholders.
- Multiple Viewpoints.
- Involves Functional and Non-Functional Requirements.
- Dynamic Environment.
- Initial requirements may be dropped.
  - To resolve requirement inconsistencies.
  - To resolve implementation inconsistencies.
  - To resolve design inconsistencies.
AGENT’S Contributions

- **Autonomous**
- **Pro-active**: able to actively pursue its goals.
- **Reactive**: able to perceive its environment and react in a timely fashion to changes that occur in order to satisfy its goals.
- **Social capabilities**: capable of interacting with other agents (humans) in order to satisfy its goals.
- **Learning**: able to utilise knowledge gained from past events to solve current problems so as to achieve its goals.
- **Mobility**: able to move from system to system
REAGENT: A living system.

Each stakeholder is assigned an agent. Agents in the REAGENT perform these tasks: *Monitoring of Specification and Environment.*

- Specifications are monitored through the development cycle by agents for opportunities to suggest repair strategies where requirements that were previously dropped due to inconsistencies can be reintroduced.
- The runtime environment is monitored to check that requirements are being satisfied and repair strategies (how dropped requirements can be modified and reintroduced) are suggested.
- REAGENT represents a *live requirements* system – requirements remain *live* throughout the lifetime of a system. Agents look out for the interests of user requirements at all times.
REAGENT at a glimpse.

- Agents assist in requirement elicitation, inconsistencies and conflict resolution.
- Agents are used as modelling constructs.
- Agents are executed as implementations.
- Agents monitor development constantly to satisfice stakeholders’ requirements.
- Agents monitor for changes in the runtime/operational environment.

Automated Change Update

Analysis

Develop / Implement

Repair / Maintain
REAGENT: Agent Capabilities

- Agent
  - Requirements Repository
  - Code Generator
  - Requirements Consistency Checker
  - Conflict Resolution Capabilities
  - Change Monitor

- Elicitation Interface
- Stakeholder

- Executable Specification
- Runtime Environment
- Conflict Resolution Server
Agents represent the interest of stakeholders when resolving inconsistencies and conflicts.

Agents participate in negotiation sessions in order to generate a consistent requirements specification.

Agents represents the interest of stakeholders when resolving inconsistencies and conflicts.
REAGENT: Implementation Phase

- Requirement are converted to executable specification.
- Executable specification are usually agents.
REAGENT: Maintenance Phase

- Change Monitors detect system change and modify specification as required.
- Stakeholders can add/remove requirements.
Conclusion

- Agent technologies offer a unique way to address software engineering problems.
- A quote by E.V.Berard, “Walking on water and developing software from a specification are easy if they are frozen.
- With REAGENT a specification need not be frozen.

As we look forward to the new millennium, agent technologies will play an even more important role as more software engineering processes are automated.