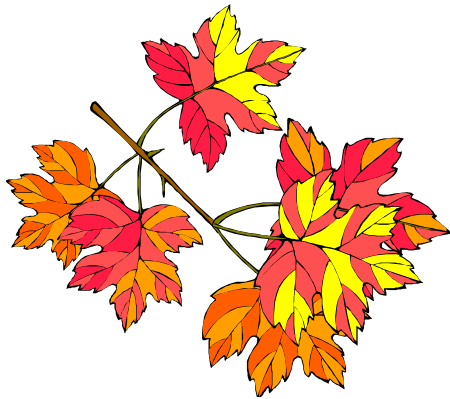
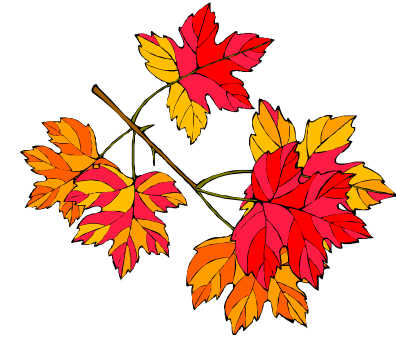


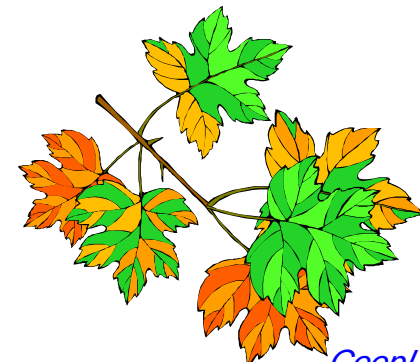
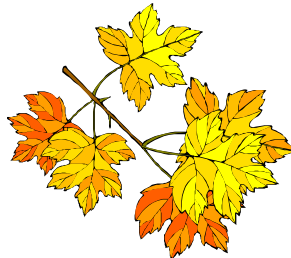
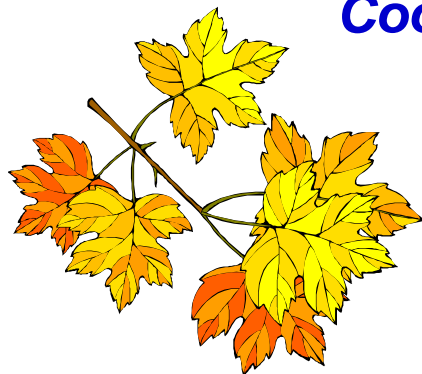
Ten Years of CoopIS Conferences: Cooperative Information Systems Then and Now



**John Mylopoulos
University of Toronto**



**11th IFCIS Conference on
Cooperative Information Systems (CoopIS'03)
November 5-7, 2003, Catania**



Abstract

- *The idea of focusing research on a new type of information system was launched with a series of workshops, held in Sydney, Niagara-on-the-Lake, Como, and Dagstuhl. This idea gained a foothold in the research establishment with the First International Conference on (Intelligent and) Cooperative Information Systems (CoopIS for short) held in Rotterdam in 1993, and never looked back since then.*
- *We review some of the early proposals put forward in those early days, and how these have evolved over the years. We also outline our current work, which assumes that cooperative information systems of the future will be agent-oriented software systems combining some of the features of multi-agent systems in AI with methodological principles from Software Engineering.*

Ten Years of CoopIS Conferences

- *The First International Conference on Cooperative and Intelligent Information Systems was held in Rotterdam in May 1993, with Mike Papazoglou, Mike Huhns and Gunter Schlageter as key organizers.*
- *It was preceded by several workshops held at IJCAI'91 (Papazoglou), Niagara-on-the-Lake (1991 -- Balzer, Mylopoulos), Como (1992 -- Brodie, Ceri), and Dagstuhl (1992 -- Ellis, Jarke).*
- *The conference was followed up by other like-minded venues, including the CoopIS Journal, special journal issues and collected volumes.*
- *The name (CoopIS) was coined in 1994.*

But what is a Cooperative Information System (CIS)?

- *An open, distributed information system that interoperates with other systems within an organizational context (syntactic and semantic interoperation) and contributes to the fulfillment of their mandate (cooperation).*
- *There are three facets to CIS: a distributed systems facet (openness, coordination, evolution,...), a collaborative work facet (workflows, business processes, CSCW,...), and an organizational facet (business processes, strategic objectives,...) [Agostini98].*

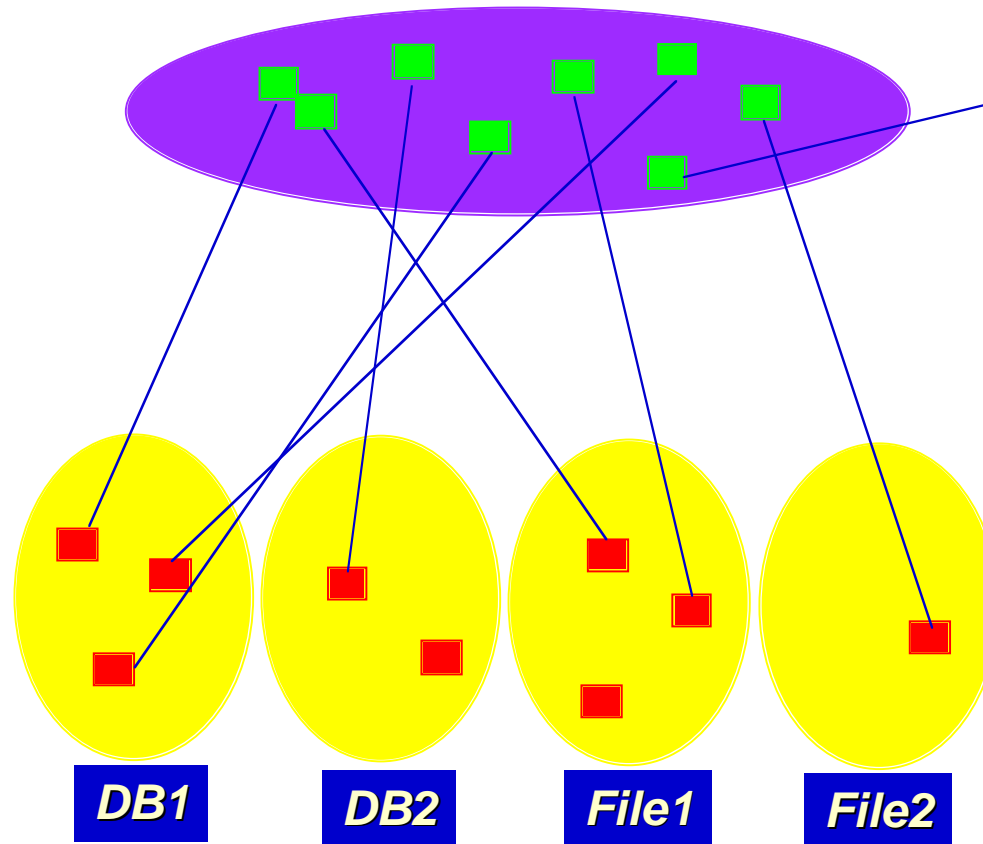
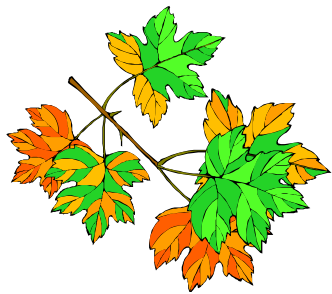
Looking into the Future -- Ten Years Ago

- *Databases will be replaced by information repositories; application software will be replaced by software agents.*
- *The shift from databases to information repositories will be caused by the need to integrate representation and implementation technologies which are currently developed independently of each other.*
- *The need for openness, evolution and distributed computing calls for agency and cooperation models; these can import results from AI planning and DAI but must come with a methodology that can "hardwire" goals and plans into conventional implementation structures.*

Information Repositories

The repository contains objects which represent other information sources (or components thereof), as well as other entities in the application domain.

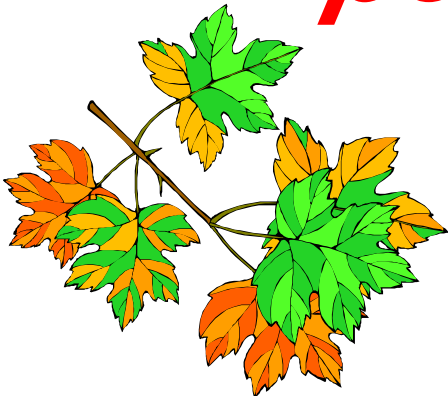
Metadatabase



...Ten Years Later...

- Tons of work on ***data integration*** (local-as-view, global-as-view,...) -- emphasis on query processing
- A lot of work on ***ontology-based*** data integration -- emphasis on semantic query processing.
- ***Data warehouses*** constitute a major data technology with a larger market share than DBMSs -- emphasis on OLAP, i.e., being able to process certain types of queries.
- WWW is ***the*** major advance in information management (actually, in all of Computer Science) in the past decade, and it was somehow missed...

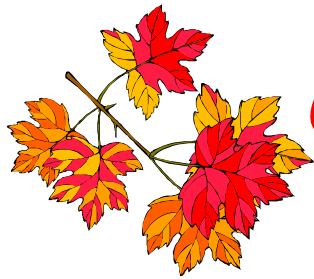
***Sometimes,
a simple idea can go a
long way, if enough
people buy into it!***



Software Agents

- *We want software methodologies that allow software to evolve architecturally and functionally, also to be distributed and open.*
- *AI planning and Multi-Agent Systems research provides a theoretical foundation for these methodologies.*
- *But we also need methodological concepts from SE; after all, they have been developing methodologies for building software for a long time.*

How do we combine the two?



Goal-Oriented Requirements Engineering (~1993)

- *Goal-oriented analysis focuses on early requirements, when alternatives are being explored and evaluated.*
- *During goal-oriented analysis, we start with initial goals such as “Higher profits”, “Every book loan request must be met” and keep refining them until we have reduced them to alternative collections of design decisions each of which can satisfy the initial goals.*
- *Initial goals may be contradictory, so the analysis must facilitate the discovery of tradeoffs and the search of the full space of alternatives, rather than a subset.*

Goal-Oriented Analysis a la KAOS

- (Organizational) goals lead to requirements.
- Goals justify and explain the presence of requirements which are not necessarily comprehensible by clients.
- Goals provide basic information for detecting and resolving conflicts that arise from multiple viewpoints [Dardenne93].
- Example goal:

SystemGoal Achieve[BookRequestSatisfied]

InstanceOf SatisfactionGoal

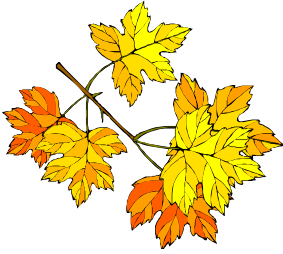
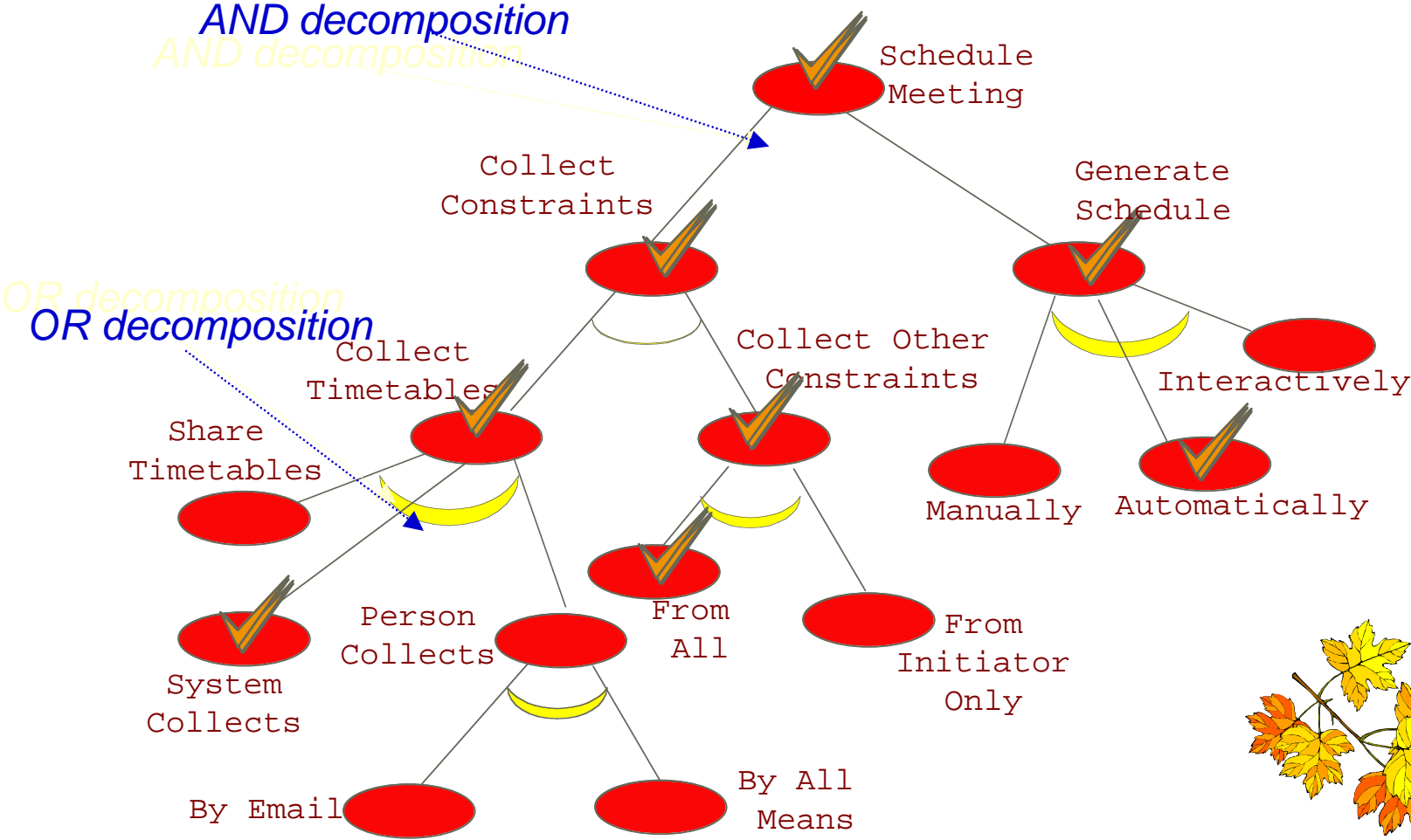
Concerns Borrower, Book, Borrowing,...

Definition ($\forall \text{bor: Borrower, } b: \text{Book, } \text{lib: Library}$)

(Requesting(bor, b) \wedge b.subject \in lib.coverageArea \Rightarrow

$\mathbf{F}[(\exists bc: \text{BookCopy}) (\text{Copy}(bc, b) \wedge \text{Borrowing}(\text{bor}, bc))]$)

Goal Analysis Leads to Alternatives

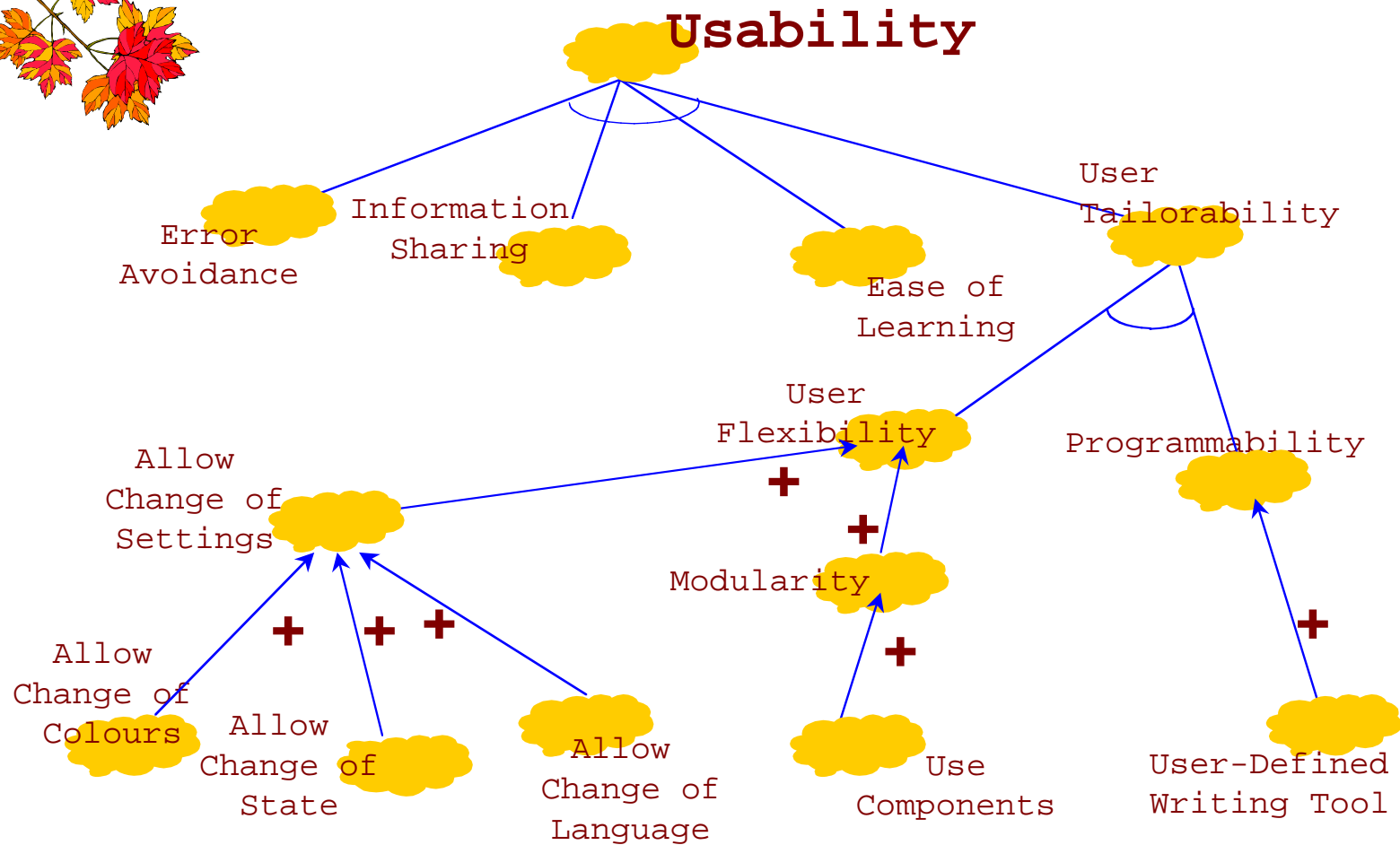
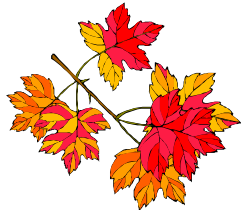


Softgoals

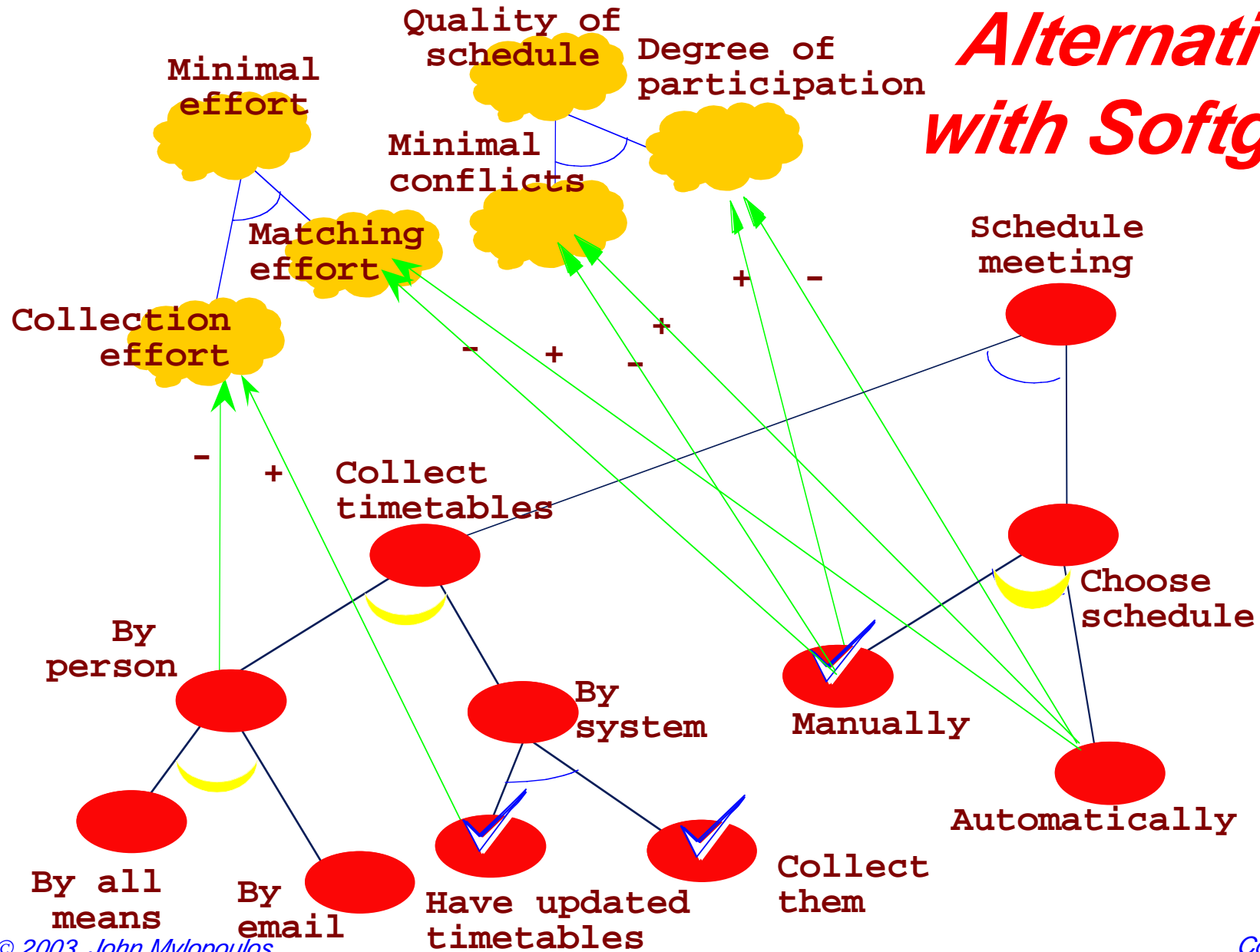
- *Functional goals, such as “Schedule Mtg” are well defined goals in the sense that there is a criterion as to whether they are satisfied or not.*
- *Non-functional goals, such “higher profits”, “higher customer satisfaction” or “easily maintainable system” specify qualities the software system should adhere to.*
- *Such qualities usually admit no generally agreed upon definition, are inter-related and often contradictory.*
- *Such goals are represented as **softgoals**.*
- *Softgoals can be thought as “fuzzy goals” with no clear-cut criteria for satisfaction; hence softgoals are **satisficed**, not satisfied.*



Softgoals for Representing Non-Functional Requirements



Evaluating Alternatives with Softgoals



Goals in Software Design

- *Both KAOS and the NFR proposal advocate the use of goals in designing software.*
- *KAOS uses goals to go from organizational objectives to functional requirements.*
- *NFR uses them to represent and analyze non-functional requirements. Non-functional requirements can be used to evaluate alternatives.*

...But are goals and goal analysis only relevant during requirements analysis?

...An Idea... (~2000)

- *Software Engineering methodologies have traditionally come about in a “late-to-early” phase (or, “downstream-to-upstream”) fashion.*
- *In particular, Structured Programming preceded Structured Analysis and Design; likewise, Object-Oriented Programming preceded Object-Oriented Design and Analysis.*
- *In both cases, programming concepts were projected upstream to dictate how designs and requirements are conceived.*

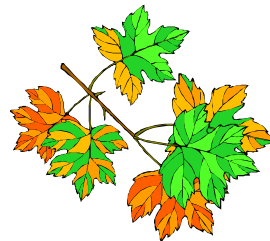
What would happen if we projected requirements concepts downstream to define software designs and even implementations?

The Tropos Project

- *Proposes a set of primitive concepts (actor, goal, actor dependency) and a methodology for agent-oriented software engineering.*
- *Covers four phases of software development:*
 - ✓ ***Early requirements** -- identify stakeholders and their goals;*
 - ✓ ***Late requirements** -- introduce system-to-be as another actor who can accommodate some of these goals;*
 - ✓ ***Architectural design** -- more system actors are added and are assigned responsibilities;*
 - ✓ ***Detailed design** -- complete the specification of system actors.*

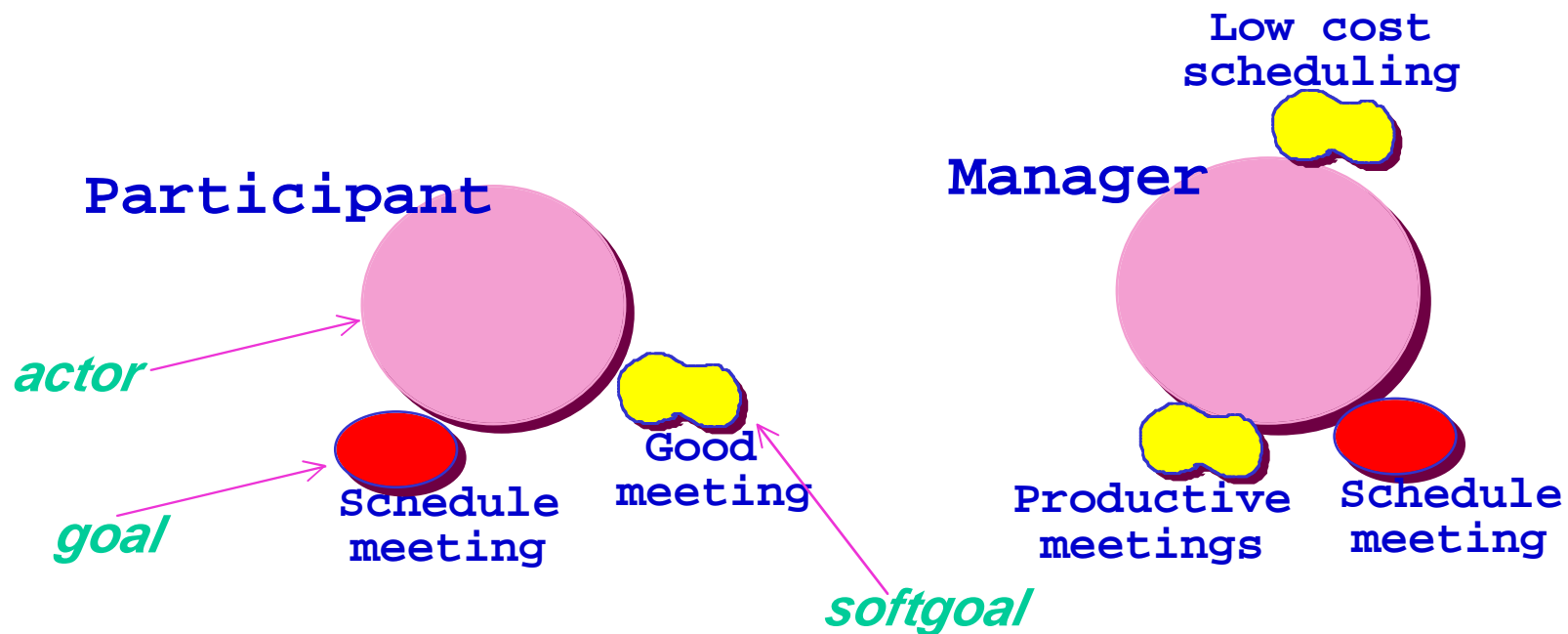
Agent-Oriented Software Engineering

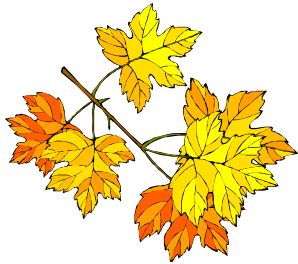
- *Many researchers working on it.*
- *Research on the topic generally comes in two flavours:*
 - ✓ *Extend UML to support agent communication, negotiation etc. (e.g., [Bauer99, Odell00]);*
 - ✓ *Extend current agent programming platforms (e.g., JACK) to support not just programming but also design activities [Jennings00].*
- *All AOSE methodologies involve to a greater or lesser extent intentional concepts, analysis of alternatives, etc.*



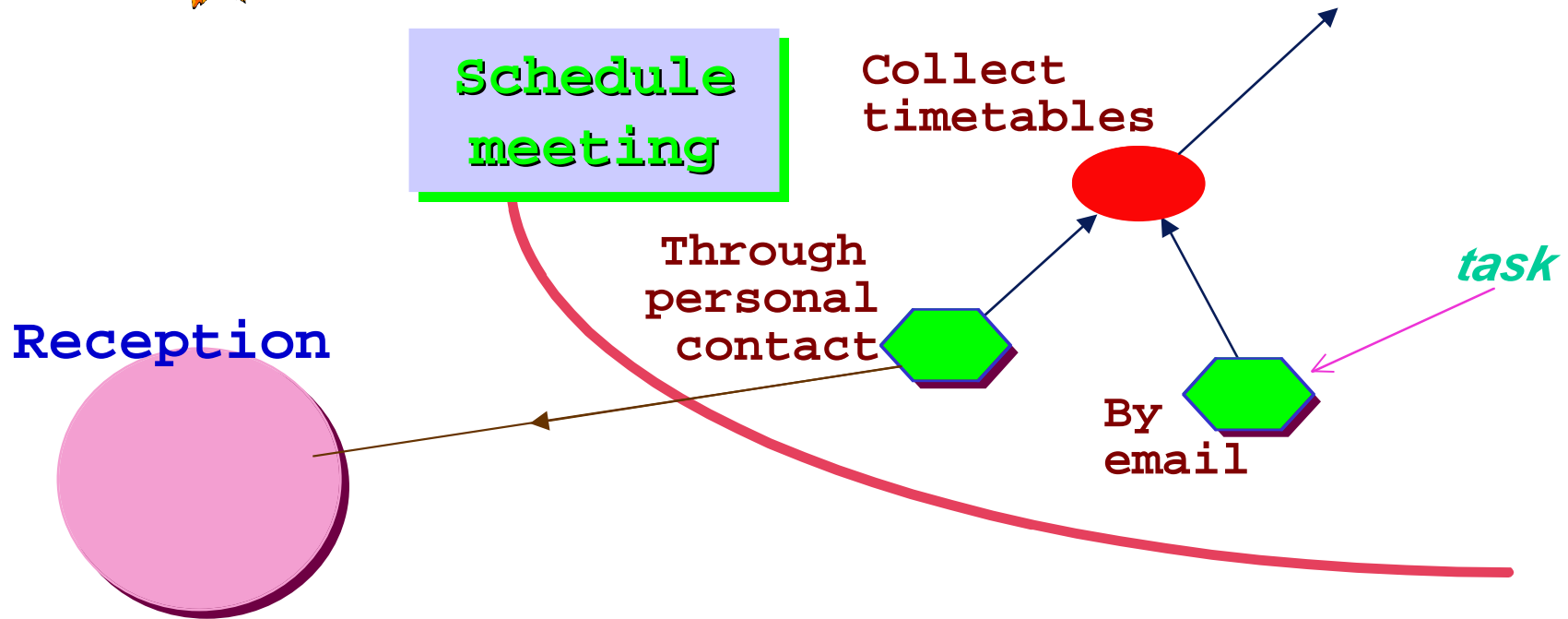
Early Requirements in Tropos: External Actors and their Goals

A social setting consists of actors, each having *goals* (and/or *softgoals*) to be fulfilled.



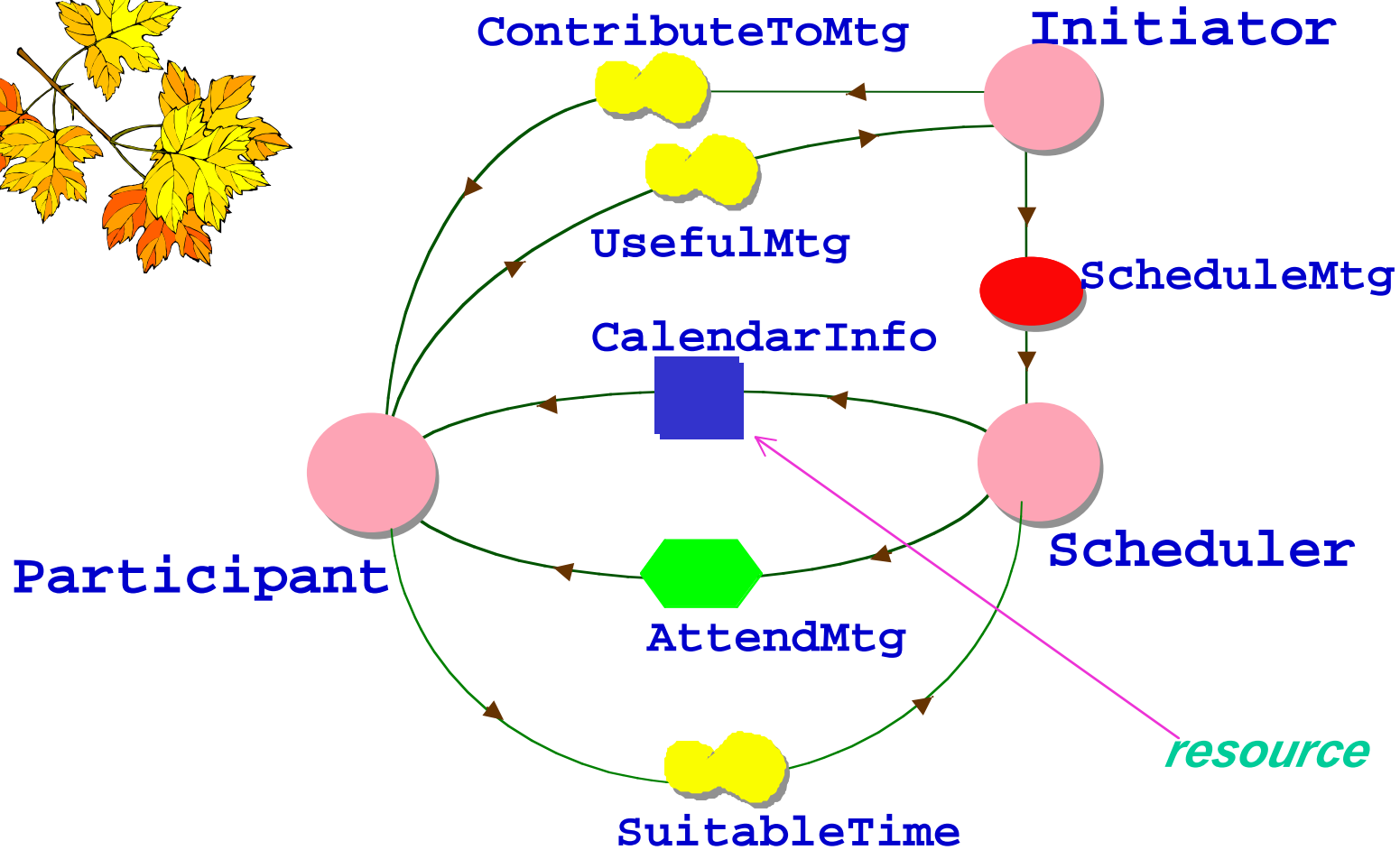
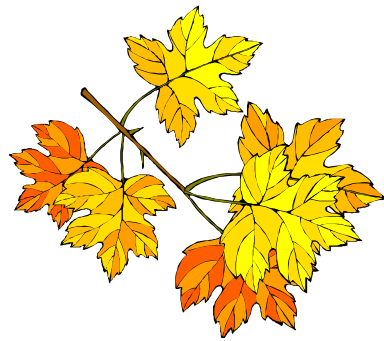


Actor Dependencies

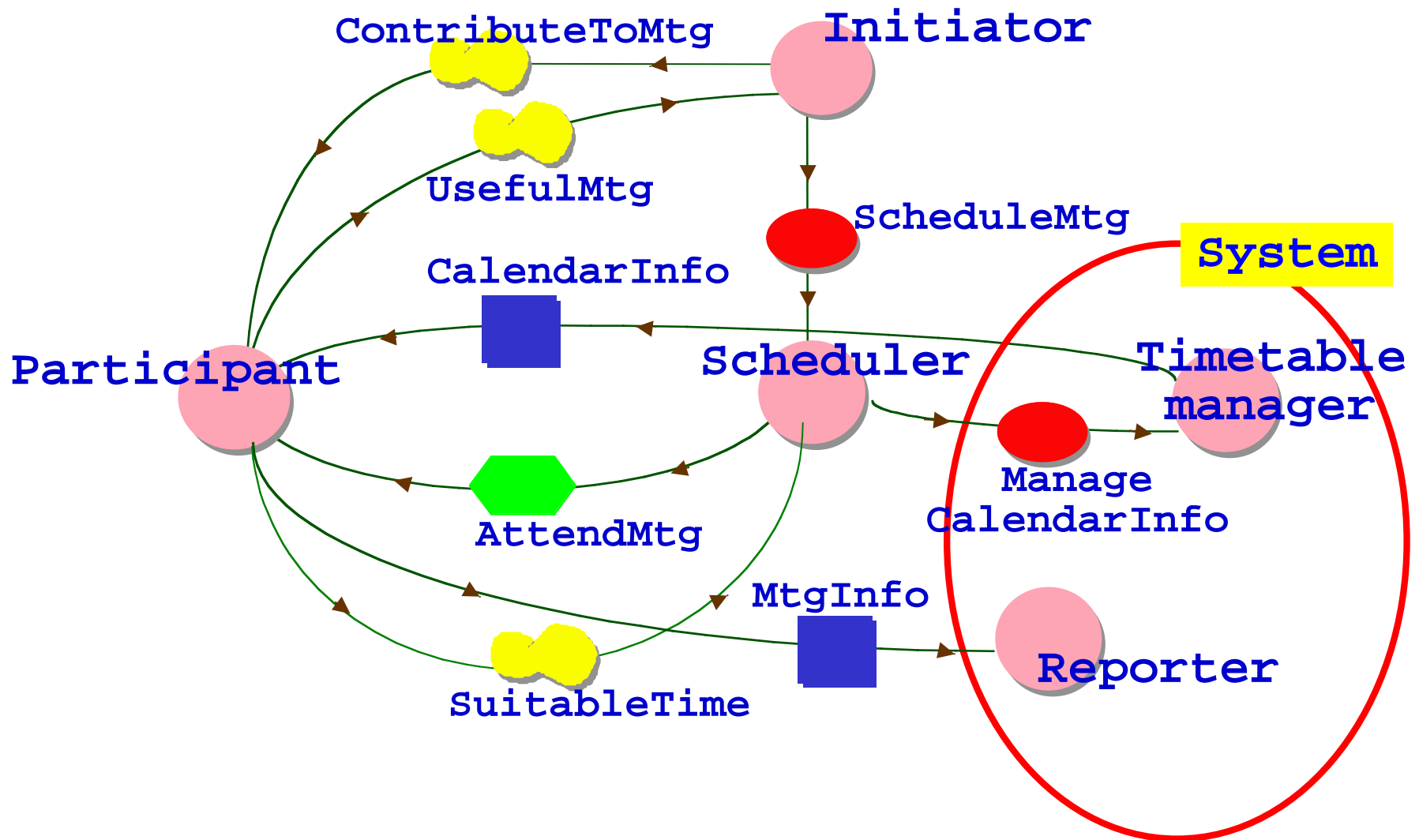


Actor dependencies are intentional: One actor **wants** something, another is **willing** and **able** to deliver.

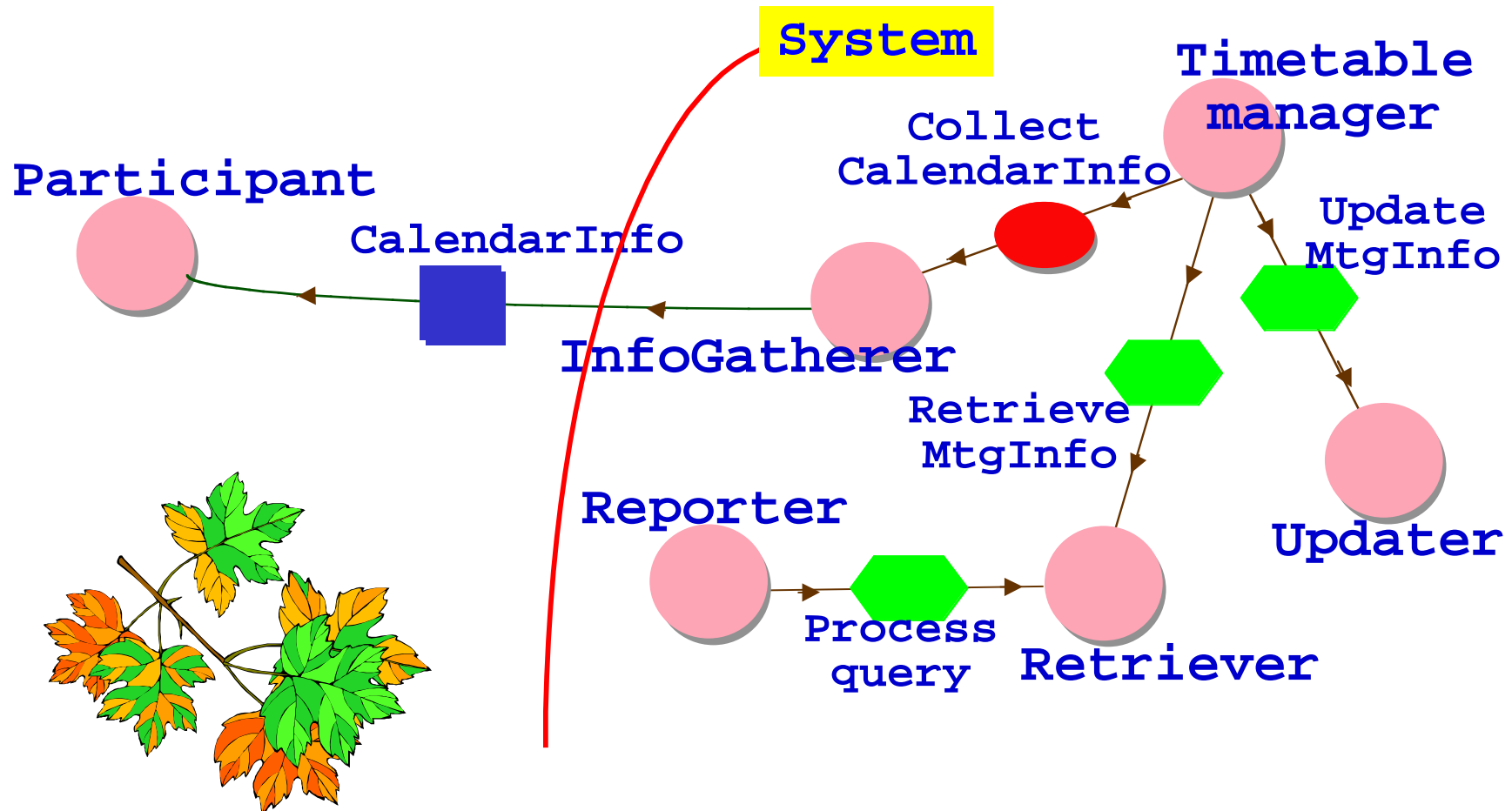
Actor Dependency Models



Late Requirements with i^*

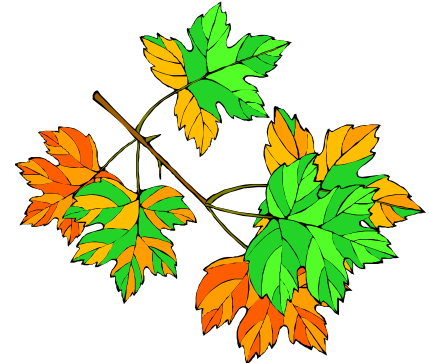


Software Architectures with *i**



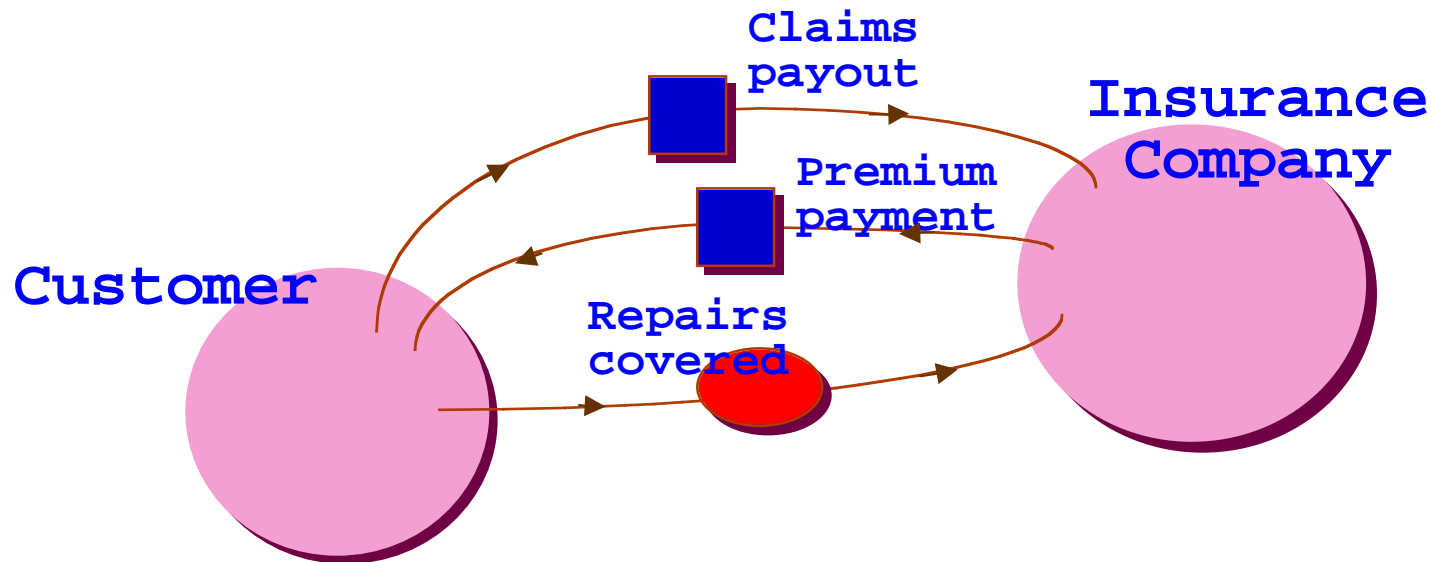
... Yet Another Software Development Process

- **Initialization:** *Identify stakeholder actors and their goals;*
- **Step:** *For each new goal, the actor who owns it:*
 - ✓ *adopts it;*
 - ✓ *delegates it to another existing actor;*
 - ✓ *delegates it to a new actor;*
 - ✓ *decomposes it into new subgoals;*
 - ✓ *declares the goal “denied”.*
- **Termination condition:** *All initial goals have been fulfilled (...to an acceptable degree), assuming all actors deliver on their commitments.*



Formal Tropos

- Each concept in a Tropos diagram can be defined formally, in terms of a temporal logic inspired by KAOS.
- Actors, goals, actions, entities, relationships are described statically and dynamically.



A Formal Tropos Example

Entity Claim

Has claimId: Number, insP: InsPolicy,
claimDate, date: Date, details: Text

Necessary date before insP.expDate

Necessary $(\forall x)(\text{Claim}(x) \wedge \bullet \neg \text{Claim}(x) \Rightarrow$
 $\neg \text{RunsOK}(x.\text{insP}.\text{car}))$

end Claim

Action MakeRepair

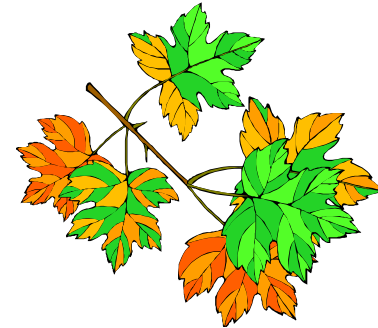
Performed by BodyShop

Refines RepairCar

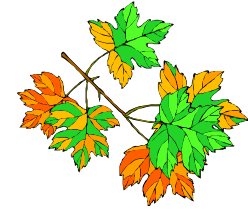
Input cl : Claim

Pre $\neg \text{RunsOK}(cl.\text{insP}.\text{car})$

Post $\text{RunsOK}(cl.\text{insP}.\text{car}) \dots$



Analyzing Models



- *Models are used primarily for human communication*
- *But, this is not enough! Large models can be hard to understand, or take seriously!*
- *We need analysis techniques which offer evidence that a model makes sense:*
 - ✓ ***Simulation** through model checking, to explore the properties of goals, entities, etc. over their lifetime;*
 - ✓ ***Goal analysis** which determine the fulfillment of a goal, given information about related goals;*
 - ✓ ***Social analysis** which looks at viability, workability,... for a configuration of social dependencies.*

Important Concepts in Goal Analysis

- An **alternative** (solution) to the fulfillment of a goal G consists of one or more leaf goals which together fulfill the root goal.
- A **goal model** defines a space of alternatives for the fulfillment of its root goal.
- An alternative A_1 is **better than A_2 in fulfilling goal G** with respect to softgoals G_1, G_2, \dots if A_1 's net contributions to G_1, G_2, \dots (e.g., positive minus negative contributions) is greater than that of A_2 .
- In general, goals and softgoals can be contradictory. Given a set of root goals and softgoals, there may not be an optimal solution [Simon68]. Hence the search for **good-enough solutions**.

A Formal Goal Model

- We use S (atisfied), D (enied) and don't assume that they are logically exclusive (remember, goals may be contradictory!)

- We offer several axioms for every goal relationship.

$$\forall g_1, g_2, g_3 [\text{AND}(\{g_1, g_2\}, g_3) \Rightarrow ((S(g_1) \wedge S(g_2)) \Rightarrow S(g_3))]$$

$$\forall g_1, g_2, g_3 [\text{OR}(\{g_1, g_2\}, g_3) \Rightarrow ((S(g_1) \vee S(g_2)) \Rightarrow S(g_3))]$$

$$\forall g_1, g_2 [++(g_1, g_2) \Rightarrow (S(g_1) \Rightarrow S(g_2))]$$

$$\forall g_1, g_2 [+(g_1, g_2) \Rightarrow \exists g [(g \neq g_2 \wedge S(g) \wedge S(g_1)) \Rightarrow S(g_2)]]$$

$$\forall g_1, g_2, g_3 [\text{AND}(\{g_1, g_2\}, g_3) \Rightarrow ((D(g_1) \vee D(g_2)) \Rightarrow D(g_3))]$$

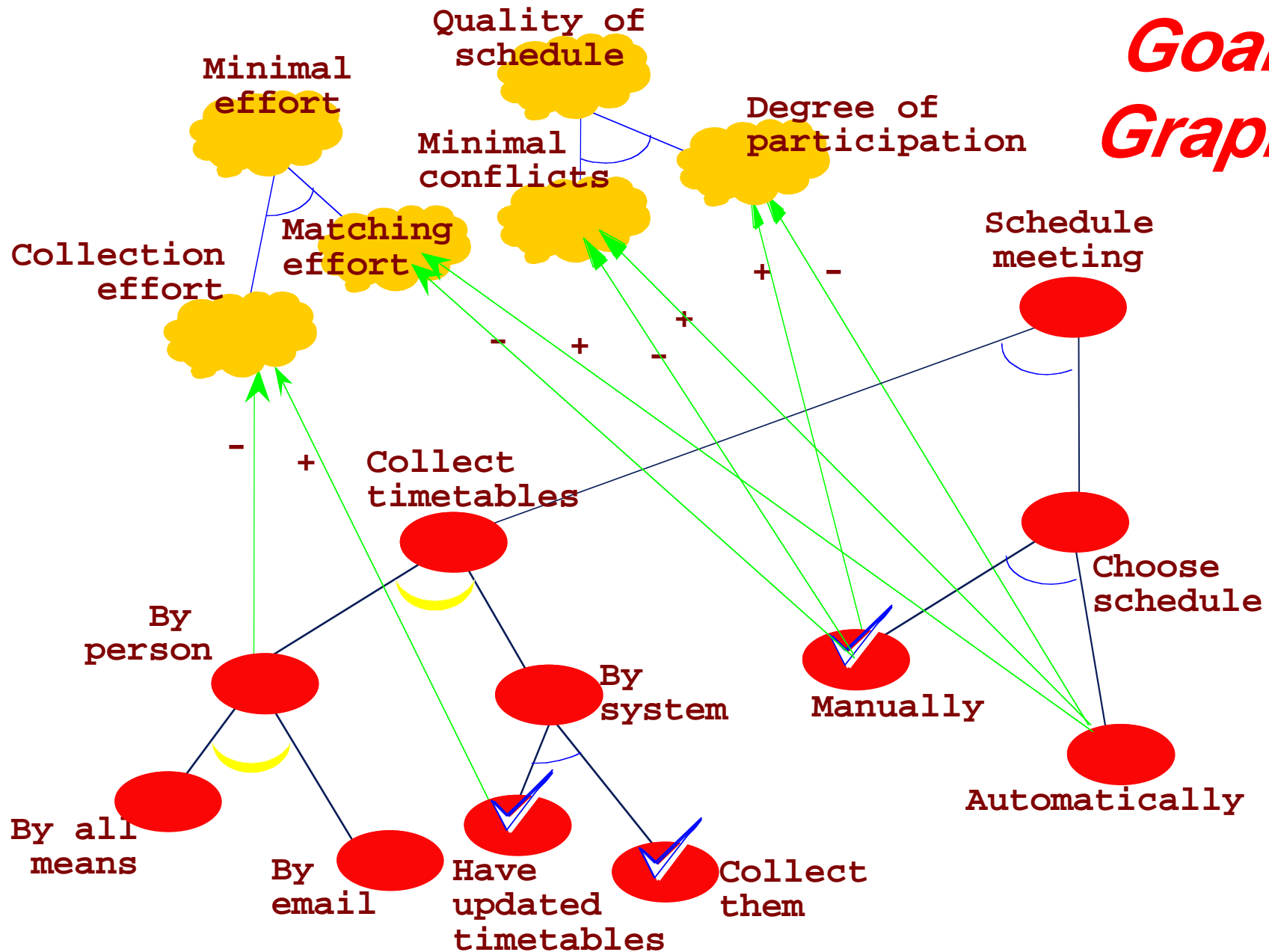
$$\forall g_1, g_2, g_3 [\text{OR}(\{g_1, g_2\}, g_3) \Rightarrow ((D(g_1) \wedge D(g_2)) \Rightarrow D(g_3))]$$

$$\forall g_1, g_2 [++(g_1, g_2) \Rightarrow (D(g_1) \Rightarrow D(g_2))]$$

$$\forall g_1, g_2 [+(g_1, g_2) \Rightarrow \exists g [(g \neq g_2 \wedge (D(g) \wedge D(g_1))) \Rightarrow D(g_2)]]$$

...more axioms for predicate D , goal relationships --, -...

Goal Graph



Qualitative Goal Analysis

- Given a goal graph, we can instantiate these axioms into a collection of propositional Horn clauses, e.g.,

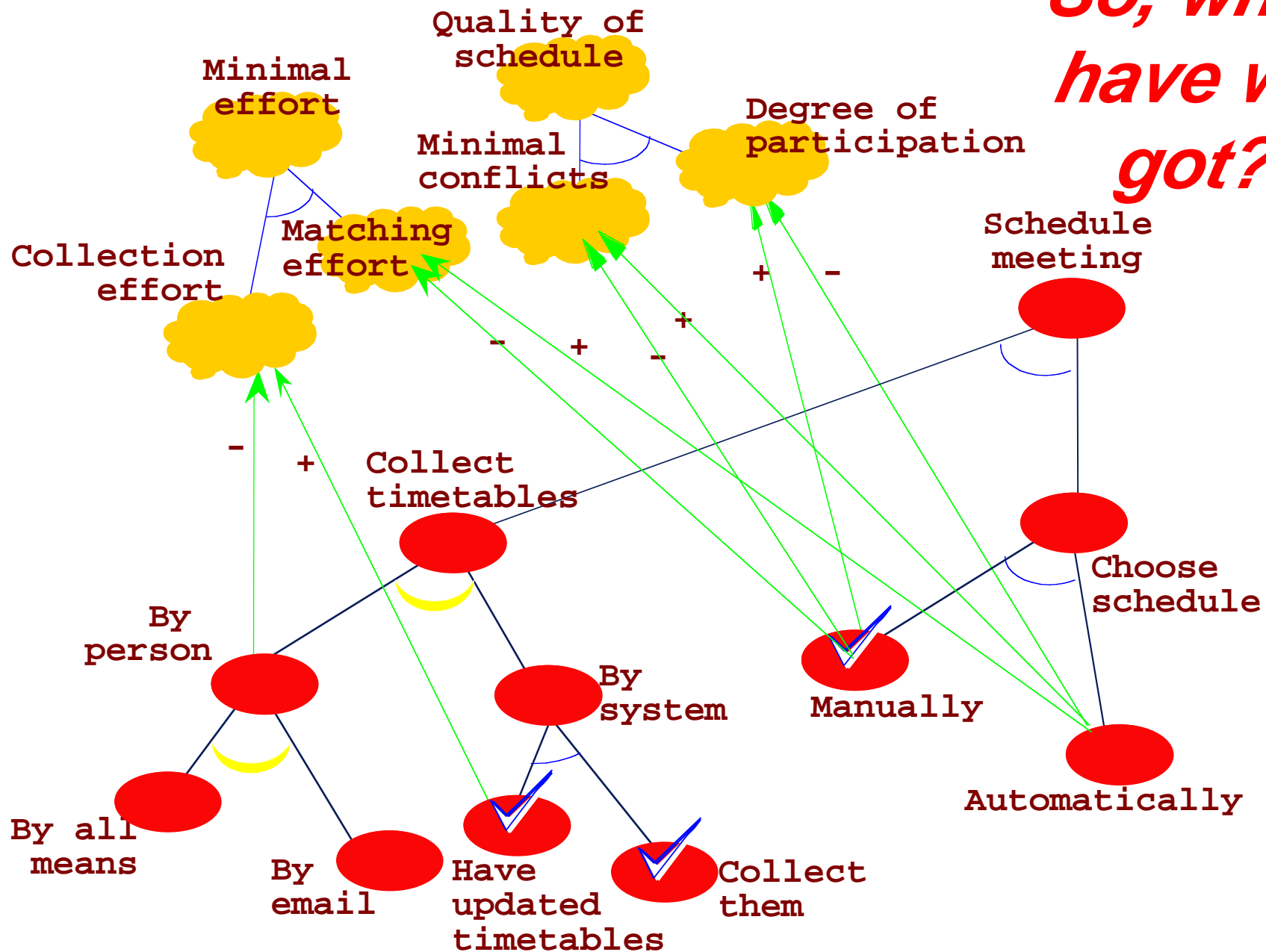
$$\forall g_1, g_2, g_3 [\text{AND}(\{g_1, g_2\}, g_3) \Rightarrow ((S(g_1) \wedge S(g_2)) \Rightarrow S(g_3))] \\ \Rightarrow (S(\text{collectTbl}) \wedge S(\text{chooseSchl})) \Rightarrow S(\text{scheduleMtg})$$

- We are also given some *S* and *D* labels for some goals, e.g., $S(\text{haveUpdatedTbl})$
- There is an $O(N)$ proof procedure which will generate all inferences from these axioms. Our proof procedure works as a label propagation algorithm.
- We have also developed algorithms to accommodate probabilities and criticalities for goals.

Bottom-Up vs Top-Down Reasoning

- *The algorithms described so far take as input a goal model and S/D labels for some of their leaf nodes and conduct bottom-up reasoning.*
- *Can we answer questions that involve top-down reasoning, such as:*
 - ✓ *Given a qualitative goal model that includes several root goals, find a solution that satisfies all of them;*
 - ✓ *Given a quantitative goal model that includes several root goals, find an optimal solution.*
- *We have solved the first problem [Sebastiani03] by reducing it to Propositional Satisfiability (SAT). The second problem can be solved with existing OR techniques.*

So, what have we got?



We are working towards a modeling framework and formal reasoning techniques that allow us to find good-enough solutions to goal satisfaction problems for problems, ranging from quantitative to qualitative, and formal to informal. This framework can be used as a basis for designing cooperative information systems.

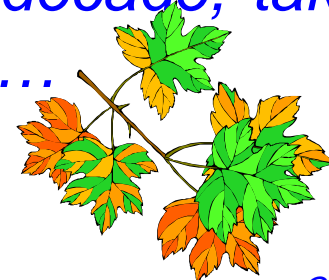
The Tropos Project

- *Project was launched in April 2000.*
- *The team of participating researchers includes:*
 - ✓ *UToronto (Canada): Ariel Fuxman, Manuel Kolp, Linda Liu, Eric Yu;*
 - ✓ *UTrento/IRST (Italy): Paolo Bresciani, Paolo Giorgini, Fausto Giunchiglia, Anna Perini, Marco Pistore, Marco Roveri, Roberto Sebastiani, Paolo Traverso;*
 - ✓ *FUPernambuco (Brazil): Jaelson Castro*
- *Publications and other information about the project can be found at*

<http://www.cs.toronto.edu/km/tropos>

Conclusions

- *Over the past decade, we have made great progress in technologies and methodologies for information management and application development.*
- *In this talk, we have focused on methodologies for application development, adopting ideas from SE and MAS. In particular, we have sketched a methodology where software design is seen as design-time planning for a multi-agent system.*
- *We expect to see such methodologies gain a foothold in SE practice over the next decade, taking over from their object-oriented cousins...*



References

- [Agostini98] Agostini, A., De Michelis, G., Jarke, M., Matthes, F., Mylopoulos, J., Pohl, K., Schmidt, J., Woo, C., Yu, E., "A Three-Faceted View of Information Systems: The Challenge of Change", *Communications of the ACM*, December 1998, 64-71.
- [Bauer99] Bauer, B., *Extending UML for the Specification of Agent Interaction Protocols*. OMG document ad/99-12-03.
- [Castro02] Castro, J., Kolp, M., Mylopoulos, J., "Towards Requirements-Driven Software Development Methodology: The Tropos Project," *Information Systems 27(2)*, Pergamon Press, June 2002, 365-389.
- [Chung00] Chung, L., Nixon, B., Yu, E., Mylopoulos, J., *Non-Functional Requirements in Software Engineering*, Kluwer Publishing, 2000.
- [Dardenne93] Dardenne, A., van Lamsweerde, A. and Fickas, S., "Goal-directed Requirements Acquisition", *Science of Computer Programming*, 20, 1993.
- [Fuxman01a] Fuxman, A., Pistore, M., Mylopoulos, J. and Traverso, P., "Model Checking Early Requirements Specifications in Tropos", *Proceedings Fifth International IEEE Symposium on Requirements Engineering*, Toronto, August 2001.
- [Fuxman01b] Fuxman, A., Giorgini, P., Kolp, M., Mylopoulos, J., "Information Systems as Social Organizations", *Proceedings International Conference on Formal Ontologies for Information Systems*, Ogunquit Maine, October 2001.

...More References...

- **[Iglesias98]** Iglesias, C., Garrijo, M. and Gonzalez, J., “A Survey of Agent-Oriented Methodologies”, *Proceedings of the 5th International Workshop on Intelligent Agents: Agent Theories, Architectures, and Languages (ATAL-98)*, Paris, France, July 1998.
- **[Jennings00]** Jennings, N. “On Agent-Based Software Engineering”, *Artificial Intelligence* 117, 2000.
- **[Mylopoulos92]** Mylopoulos, J., Chung, L. and Nixon, B., “Representing and Using Non-Functional Requirements: A Process-Oriented Approach,” *IEEE Transactions on Software Engineering* 18(6), June 1992, 483-497.
- **[Odell00]** Odell, J., Van Dyke Parunak, H. and Bernhard, B., “Representing Agent Interaction Protocols in UML”, *Proceedings 1st International Workshop on Agent-Oriented Software Engineering (AOSE00)*, Limerick, June 2000.
- **[Simon69]** Simon, H., *The Sciences of the Artificial*, The MIT Press, 1969
- **[Wooldridge00]** Wooldridge, M., Jennings, N., and Kinny, D., “The Gaia Methodology for Agent-Oriented Analysis and Design,” *Journal of Autonomous Agents and Multi-Agent Systems*, 3(3), 2000, 285–312.
- **[Yu95]** Yu, E., *Modelling Strategic Relationships for Process Reengineering*, Ph.D. thesis, Department of Computer Science, University of Toronto, 1995.
- **[Zambonelli00]** Zambonelli, F., Jennings, N., Omicini, A., and Wooldridge, M., “Agent-Oriented Software Engineering for Internet Applications,” in Omicini, A., Zambonelli, F., Klusch, M., and Tolks-Dorf R., (editors), *Coordination of Internet Agents: Models, Technologies, and Applications*, Springer-Verlag LNCS, 2000, 326–346.