

An Example

- An artist friend subscribes to AOL for email and web services.
- Recently her address book stopped working: it would pretend to start updating itself and then hang.
- She wrote to AOL explaining the problem, and they suggested a quick fix.
- She tried it and it didn't work.
- So she wrote back a nice letter saying so.
- Here was their response...

Their Response

Subj: Re: Technical Support Question
From: support@aol.net
To: HW2015@aol.com

Dear HW2015,

Hello! My name is Ghet D. Thanks for writing again to the Techmail Department of America Online.

Thanks for writing back. I apologize that the previous response did not fix the problem.

Please look at the information I have included; I hope that it will be helpful in resolving the problem this time.

NOTE: Some of our troubleshooting options below are quite lengthy and tedious to follow, so I am asking for your patience and understanding about it. However, you need not to go through all of the steps if one has already resolve the problem. Please just try the next step if problem still persist.

A. DELETE THE MAIN.IND FILE

B. REPLACE THE MAIN.IDX AND THE COMPVER.BIN FILES

C. INSTALL A NEW COPY OF AOL INTO THE EXISTING AOL DIRECTORY

A. DELETE THE MAIN.IND FILE

Deleting the main.ind file may resolve the issue.

1. Click on START, select Find, then click on FILES OR FOLDERS.
2. In the Named box, type: MAIN.IND

ensure the Look in box lists the local hard drive, normally (C:), then click on FIND NOW. Windows should find one main.ind file for each copy of AOL you have installed.

NOTE: In Windows 2000, click on Start, select Find, then click on For Files or Folders. Type in the Search for files or folders box, then click on Search now.

3. In the Name column, right-click on the MAIN.IND file, then click on DELETE.
4. Click on YES to send this file to the Recycle Bin.
5. Repeat steps 3 and 4 to delete all main.ind files, then click on the X to close the Find window.
6. Click on START, select PROGRAMS, select AMERICA ONLINE, then click on ..

AMERICA ONLINE 6.0. AOL will update to rebuild the main.ind file that was deleted.

B. REPLACE THE MAIN.IDX AND THE COMPVER.BIN FILES

Damaged copies of the MAIN.IDX and COMPVER.BIN files may cause the issue.

Please replace the MAIN.IDX and COMPVER.BIN files.

TO REPLACE THE MAIN.IDX AND THE COMPVER.BIN FILES,
PLEASE FOLLOW THE STEPS

BELOW:

1. On the Desktop, double-click on MY COMPUTER.
2. Double-click on the (C:) drive.
3. Double-click on the AMERICA ONLINE 6.0 folder.
4. Double-click on the BACKUP folder.
5. Right-click on the MAIN.IDX file, then click on COPY.
6. On the toolbar, click on the BACK button.
7. Right-click on the IDB folder, then click on PASTE.

ECSA, Sept. 2007

© Garlan 2007

7

-
8. Click on YES to overwrite the existing files.

9. Right-click on the COMPVER.BIN file, then click on DELETE.

10. Click on YES to send this file to the Recycle bin.

11. Click on the X to close the America Online 6.0 window, then repeat on all open windows.

C. INSTALL A NEW COPY OF AOL INTO THE EXISTING AOL DIRECTORY

Damaged software may cause this issue. Please install a new copy of AOL into the existing AOL directory.

NOTE: You may obtain a free copy of the AOL CD from many local retailers, including Circuit City, Barnes and Noble, Office Depot, and Office Max.

Please call ahead and ask if they have an AOL CD available for you.

ECSA, Sept. 2007

© Garlan 2007

8

To download a copy of AOL, please visit Keyword:
UPGRADE

On the Internet, please visit the Web address below:
<http://www.aol.com>

To order a CD, please call the number below:
1-888-265-8002

TO INSTALL A NEW COPY OF AOL INTO THE EXISTING AOL
DIRECTORY, PLEASE FOLLOW
THE STEPS BELOW:

NOTE: Before beginning the installation, please close all other programs,
including other versions of AOL. Click on Program on the Task Bar, then
click on the X in the upper right corner of the program's window to close
it.

ECSA, Sept. 2007

© Garlan 2007

9

1. Place the AOL 6.0 CD in the CD-ROM drive.

NOTE: If the installation starts automatically, skip to step 5 below. If the
installation does not start automatically, or if you are installing from a
download rather than the CD, follow all the steps below.

2. Click on START, select Find, then click on FILES OR FOLDERS.

NOTE: In Windows 2000/Me, click on Start, select Search, then click on
For Files or Folders.

3. In the Named box, type
SETUP*.EXE

Ensure the Look in box lists the local hard drive, normally (C:), then click
on FIND NOW.

NOTE: In Windows 2000/Me, type in the Search for files or folders named
box, then click on Search Now.

ECSA, Sept. 2007

© Garlan 2007

10

NOTE: If you are installing AOL from the CD, change the Look in box to show the CD-ROM drive, normally (D:), then click on Find Now or Search Now.

4. Double-click on the SETUP file that has an AOL icon (a white triangle on a blue background).
5. Click on OK if you receive a message recommending that you exit all other applications.
6. Click on CURRENT MEMBERS. AOL will search for other versions. This might take a moment.
7. Select Upgrading to a new version of AOL on this computer, then click on NEXT.
8. Click on YES to install the same version.
9. Select the most current Version of AOL, write down the directory, then click on NEXT.

-
10. Select Copy the files to the new version of AOL, then click on NEXT.

NOTE: If you do not have any downloaded files, the Select Downloaded Files Action window will not appear. Skip to step 11.

11. Click on CLICK HERE.
12. Click in the box at the top, press the END key on your keyboard, then press the BACKSPACE key to delete the last letter of the directory. If necessary, type a Letter at the end so the directory matches what you Wrote down, then click on DONE.
13. Click on NEXT. The installation will run, then AOL will launch automatically. This may take a few minutes.

It has been my pleasure re-assisting you.

Should you have further concerns or queries, please feel free to write back and I will be more than happy to provide re-assistance to you. Or, visit us at Member Help Interactive by going to keyword: ASK THE STAFF and select GET LIVE ONLINE HELP and then choose TECHNICAL so we can assist you interactively.

Take care and have a nice day!

Very truly yours,

Ghet D.
Customer Care Consultant
Techmail Department

Her Response

- From: HW@aol.com
- To: support@aol.net

Dear AOL:

Are you out of your mind? Perhaps I can find someone who has the patience and knowledge to do all the 400 steps that you suggest. Is there any way that it can be fixed on your end? I can't believe your suggestion. **Why don't you get a home rocket building kit and send a rocket to Mars! Just follow the directions carefully.**

This Talk

- The Problem with Today's Computing
 - User distraction from information and technology overload
- Task-Oriented Computing
 - Raising the level at which computers help people
- Two Examples
 - Personal Cognitive Assistance on the Desktop
 - Handling Mobility and Resource Variability for Ubiquitous Computing
- Common Themes and Architectural Research Challenges

The Problem

- Computer-mediated activities are now a central part of our daily lives
 - Communication, scheduling, planning, purchasing, accessing information, ...
- Today's systems provide weak support for helping us with our high-level tasks
 - Require that we map our high-level goals into low-level, application-specific commands and actions
 - Require that we understand details of technology
- Consumes more and more of our time to manage technology and information
 - And it is getting worse all the time

Solution: Task-Oriented Computing*

- Support user intentions
 - capture high-level intent as tasks
 - raise level of abstraction of user interactions
 - automatically handle routine tasks
- Support mobility and environment variability
 - suspend/resume on different platforms and locations
 - dynamically reconfigure to match available resources
 - adapt to faults, changing user needs
- Support proactivity
 - active guidance and assistance from the system
 - corrections, work-arounds, anticipatory configuration

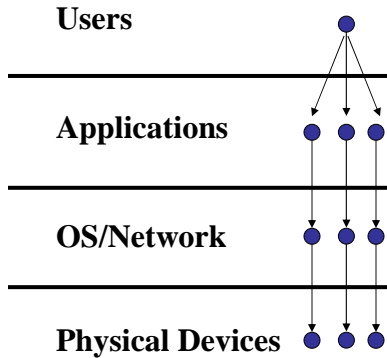
* Sometimes called “activity-oriented computing”

Tasks

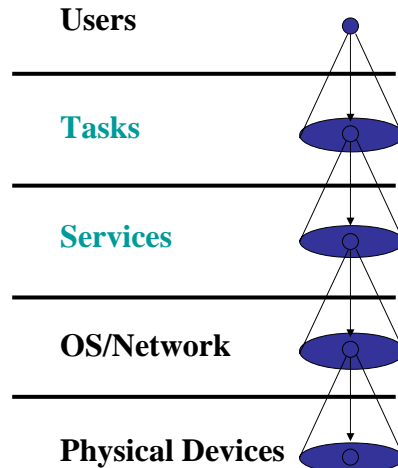
- Tasks encode user goals
 - **In office:** preparing a lecture, writing a report, managing calendars, keeping web sites up-to-date
 - **In home:** cooking dinner, relaxing in evening, support home security
 - **In car:** planning a trip, being entertained
 - **In commercial environment:** adapting system to repair problems, deal with security attacks, improve performance

A New Layer of System Structure Needed

Today



Task-oriented Computing



ECSA, Sept. 2007

© Garlan 2007

19

Two Examples

- **RADAR**
 - A Personal Cognitive Assistant
 - DARPA-funded project to put “learning on the desktop”
 - Empirical demonstration of usefulness
- **Aura**
 - Distraction-free Ubiquitous Computing
 - Focus on heterogeneous hardware and mobility
 - Applications to smart environments (smart office, smart home, smart car, ...)

ECSA, Sept. 2007

© Garlan 2007

20

RADAR: The Vision

- A **Personal Cognitive Assistant (PCA)**
 - Like a good secretary
 - Helps us with routine tasks, giving us more time for fun/creative/challenging activities
 - Understands our needs and preferences
 - Adapts to our behavior over time
 - Helps where we desire it, but gets out of the way otherwise
 - Asks questions when not sure how to proceed
 - Does not require us to change the way we like to do business

Example

- **Professor Kim sends email** saying he will be visiting CMU and wants to meet with me and give a research talk.
- My PCA reads the message and realizes that there are several automatable tasks involved
 - **Reserves time on my calendar** for a meeting with Kim
 - **Locates a lecture room**, and **reserves** it for Kim's talk, making sure the room has sufficient equipment
 - **Updates the departmental web site** after **requesting a title, abstract and bio by email** from Kim
- Along the way it confirms various actions with me to be sure that it is on the right track.

The RADAR Project

- RADAR = “Reflective Agents with Distributive Adaptive Reasoning”
- About 40 researchers (faculty, staff, students)
- DARPA-supported (under PAL Program)
- Focus on Learning
- Now starting its fourth year
- Stringent, real-world evaluation

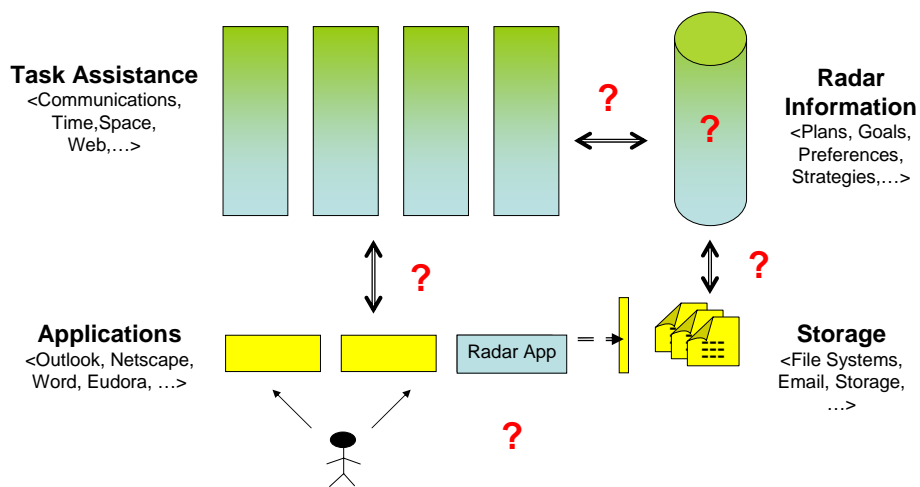
The RADAR Architecture for
Personal Cognitive Assistance
Garlan, Schmerl
*International Journal of Software
Engineering and Knowledge
Engineering, April 2007.*

ECSA, Sept. 2007

© Garlan 2007

23

The Radar System Concept



ECSA, Sept. 2007

© Garlan 2007

24

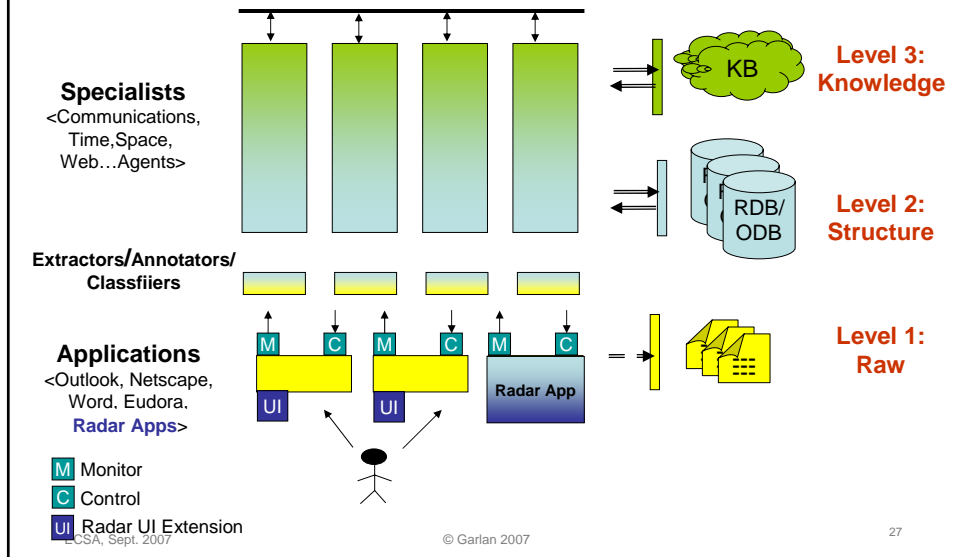
Key Architectural Drivers

- **Compatibility**
 - Must work with existing applications, information, processes, interfaces, policies
- **Extensibility**
 - Must be able to incrementally add new capabilities to support new kinds of tasks
- **Adaptability**
 - Must adapt to individual user's needs and preferences over time
 - Must become more useful and helpful over time

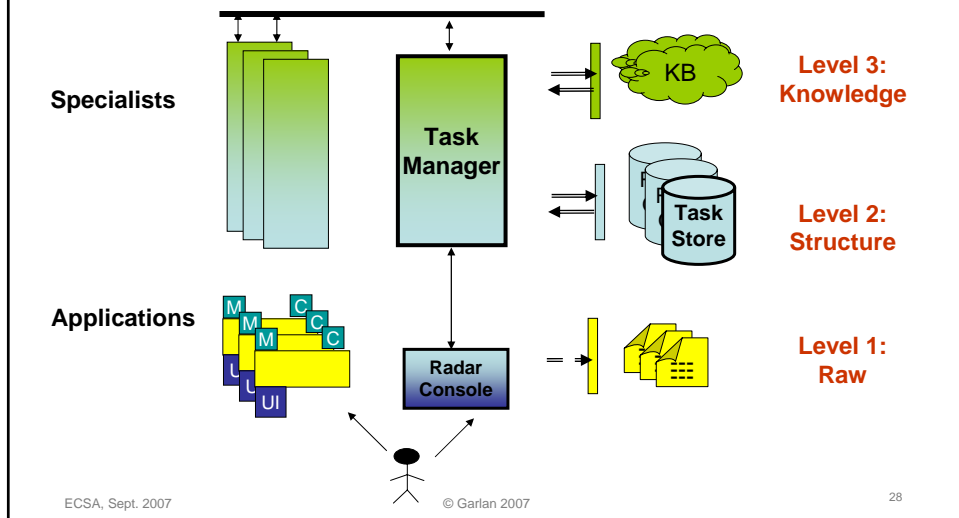
Other Drivers

- **Reliability and Availability**
 - Comparable to email systems
- **Security and privacy**
 - Access to personal information should be controlled
- **Scalability**
 - Should support hundreds of users

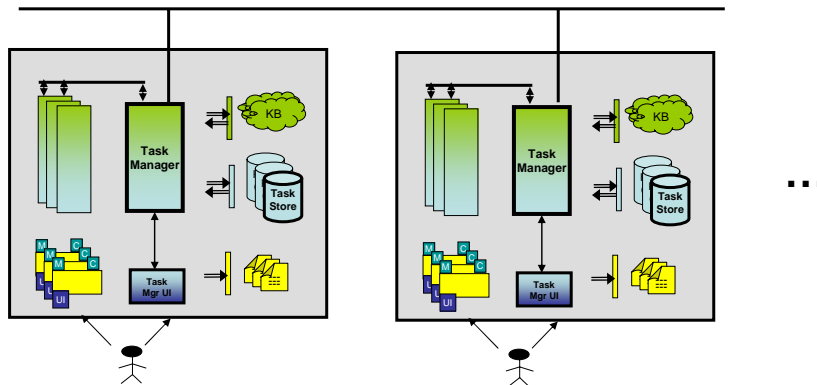
The Radar Architecture



Task Coordination (single user)



Task Coordination (multiple users)



ECSA, Sept. 2007

© Garlan 2007

29

Radar Architecture Concepts

- Key concepts
 - **Task Specialists** carry out various kinds of task assistance, like calendar management, web updating, etc.
 - The **Task Manager** coordinates specialists, and provides other shared services (task dispatching, event notification, task history and status)
 - **Categorizers** and **Classifiers** perform NLP on email messages to support automated processing
 - A **Knowledge Base** stores semantically rich information and relationships learned by RADAR

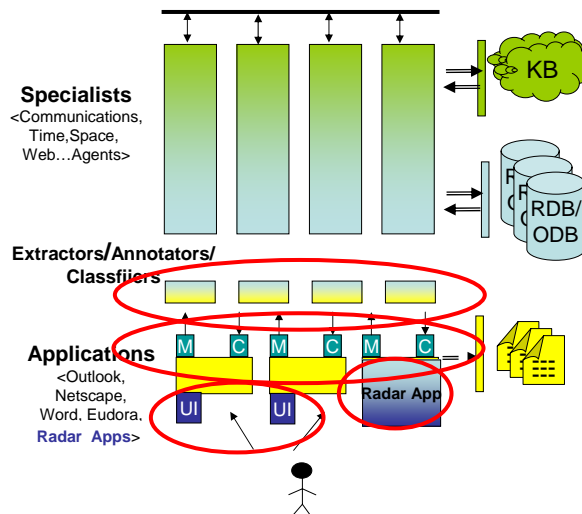
ECSA, Sept. 2007

© Garlan 2007

30

Compatibility

- Use standard application APIs and wrappers
- Use classifiers, extractors with NLP to bring legacy information and events into “task space”
- Use client-side interface technology (e.g., browsers, Outlook) to interact with user



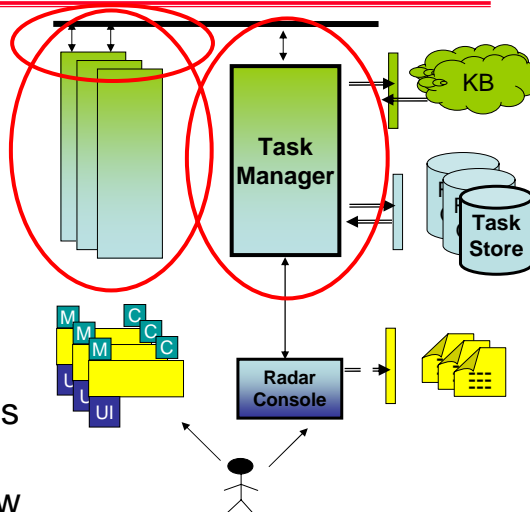
ECSA, Sept. 2007

© Garlan 2007

31

Extensibility

- Plug-in architecture allows new “specialists” to be added incrementally
- Loose coupling between specialists promotes reusability
- Centralized task management services simplify construction and integration of new specialists



ECSA, Sept. 2007

© Garlan 2007

32

Adaptability

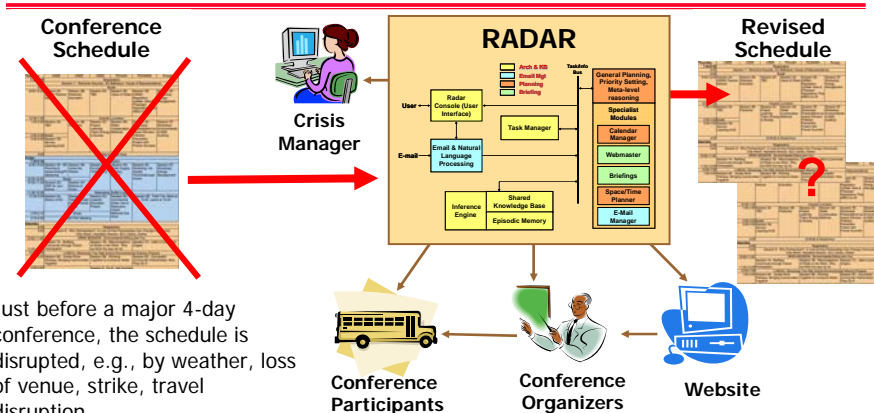
- Learning is built in to everything
 - Use common learning packages (e.g., MinorThird)
- Mixed-mode initiative allows RADAR to cooperate with user
 - Adjusting degree of automation over time
- Examples of learning and adaptation
 - Email classification and feature extraction
 - Social net knowledge
 - Task prioritization
 - Task identification and dependencies
 - Calendar preferences

ECSA, Sept. 2007

© Garlan 2007

33

RADAR Conference Planning Task



Just before a major 4-day conference, the schedule is disrupted, e.g., by weather, loss of venue, strike, travel disruption...

The user (and RADAR) must assess the situation, develop a revised plan, get the word out about the new plan, and deal with queries and requests during the crisis.

The results are evaluated on:

- Quality and completeness of the new plan
- Successful completion of related tasks
- Costs of the solutions

supported by www.garlan.org

© Garlan 2007

34

Current Work

- Task coordination, prioritization, and advice
- Task inference and planning
- Increased use of knowledge base
- Subsets for daily use
- Application to aircraft maintenance (with Boeing)
- Augment test to compare also with human assistant

What is Missing from RADAR?

- Support for mobility
- Heterogeneity of platform, infrastructure, devices, ...
- Awareness of context

Aura: The Vision

- Reduce user distraction in ubiquitous computing environments
- Support optimal environment configuration and reconfiguration
 - optimality conforms to preferences of user, needs of the task, and user's context
 - adapts environment when task, environment, or context changes
- Provide continuity for mobile users
 - system locates and configures the devices and services in the environment for user to move from one environment to another
 - proactively stages information anticipating user's needs

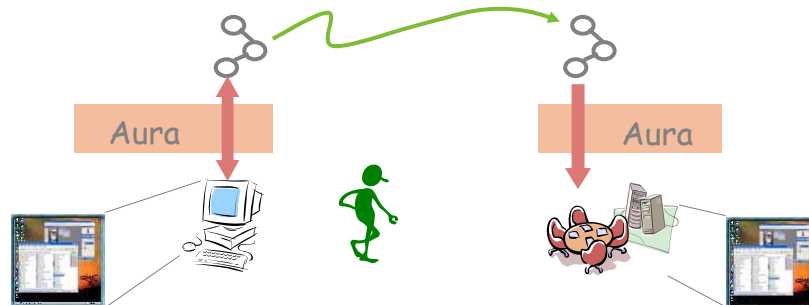
ECSA, Sept. 2007

© Garlan 2007

37

Example: Supporting User Mobility

- Going from office to home
 - Today: save files, close applications; transfer documents; restart applications on correct files; adjust settings; remember URLs, relevant emails, etc.
 - Aura: handles this automatically



ECSA, Sept. 2007

© Garlan 2007

38

Aura History

- Started in 1999
 - Initially funded by DARPA, later NSF & industry
- Many researchers
 - Operating systems, networks, HCI, wearable computing, speech, software engineering
- Initial focus: CMU campus and smart office
 - Campus location awareness
 - Desktop environments and applications
- Recent focus: smart home
 - Collaboration with ETRI

Project Aura: Towards Distraction-Free Pervasive Computing
Garlan, Siewiorek, Smailagic, Steenkiste.
IEEE Pervasive Computing 1(2):22-31
April-June 2002

ECSA, Sept. 2007

© Garlan 2007

Aura Concept

- Users define tasks as collections of abstract services
 - Edit text, view video, ...
 - Tasks can be started, paused, stopped
- Aura handles all of the details of configuration
 - Finding the right set of actual applications, services, and devices in the environment to carry out task
 - Adjusting QoS parameters to handle resource scarcity
 - Optimally configuring task to match task needs and user preferences and future predictions of need
- And reconfiguration
 - When the user moves
 - When the environment changes

ECSA, Sept. 2007

© Garlan 2007

40

Key Architectural Requirements

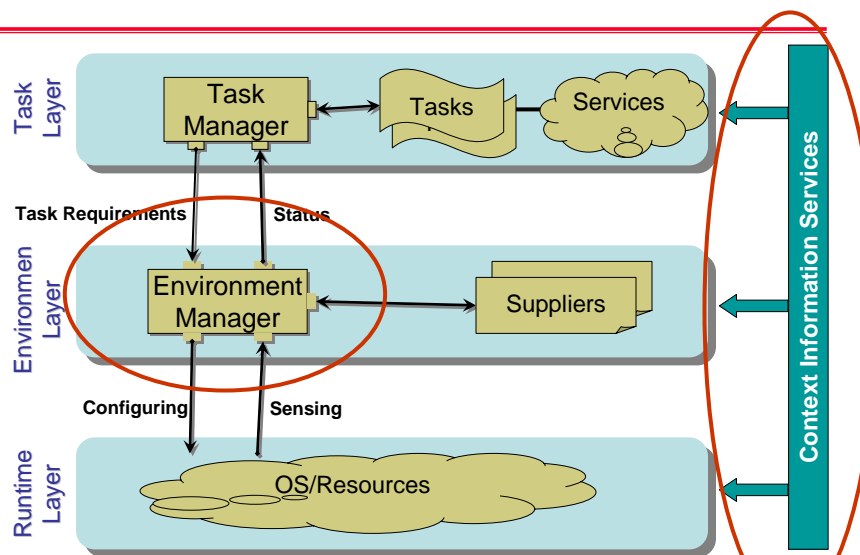
- **Compatibility**
 - Must take full advantage of existing resources in the environment (platforms, devices, sensors, actuators)
- **Extensibility**
 - Must be able to incrementally add capabilities to support new kinds of tasks, devices, resource monitors, multi-fidelity applications, resource predictors
- **Adaptability**
 - Must support dynamic reconfigurability in face of changes in task, user context, resources, future predictions

ECSA, Sept. 2007

© Garlan 2007

41

The Aura Architecture



ECSA, Sept. 2007

© Garlan 2007

Not in Radar

42

Aura Architecture Concepts

- **Task layer** keeps track of user tasks
 - Task = *Collection of abstract services and information* needed to carry out a task
 - Quality of service goals, knowledge of user preferences, task steps
- **Environment layer** carries out configuration and reconfiguration
 - Maps task requirements to available services & applications (called **suppliers**)
 - Makes tradeoffs to maximize utility for task given available services and resources, using fast algorithms
- **Context information services**
 - Provide user location, etc. to all layers

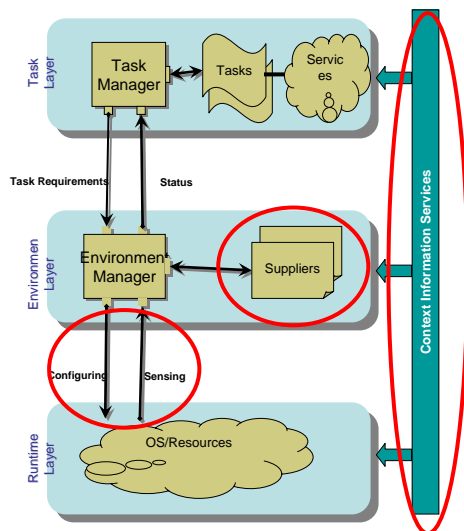
ECSA, Sept. 2007

© Garlan 2007

43

Compatibility

- Resource monitors & predictors inform Evt Mgr of current and future state
- Suppliers are wrappers around existing applications and services
- Context information services leverage existing information: calendars, sensors, wireless routers



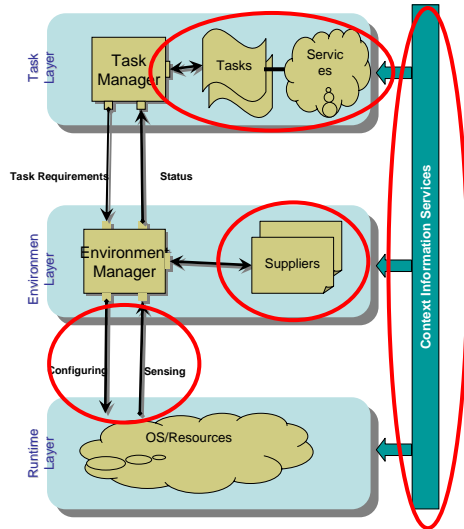
ECSA, Sept. 2007

© Garlan 2007

44

Extensibility

- User can add new tasks to system
- Can add new resource monitors & predictors
- Can add new services by wrapping existing applications and services
- Can add new context information services



ECSA, Sept. 2007

© Garlan 2007

45

Adaptability

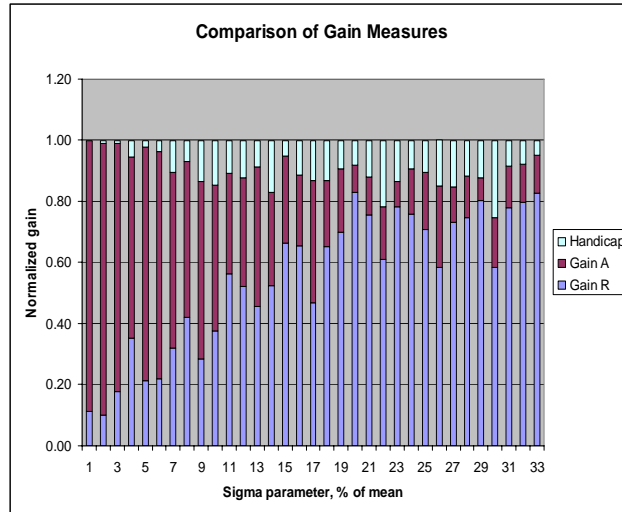
- Occurs at all levels of the system
- OS, Networks, Devices
 - Performance, anticipatory data staging
- Environment manager
 - Dynamically configures services based on available apps & resources, fidelity vs resource tradeoffs, user preferences, needs of task, context
- Task manager
 - Allows users to start and stop tasks, create new tasks, change QoS requirements for any task
- Context information services
 - Provide dynamic updates on user's context

ECSA, Sept. 2007

© Garlan 2007

46

Comparison of Utility Gain Measures



ECSA, Sept. 2007

© Garlan 2007

47

Anticipatory Algorithm Runtime Efficiency

Table 1: running times of one invocation of the algorithm, OptActUtilExpectation, in milliseconds. The row labels show the simulation depth, and the column labels show CPU speed. CPU is 100 percent utilized.

	2	3	4	5	6
Max	0.50	1.95	7.65	24.20	72.50
Slow	1.71	6.15	23.30	72.00	201.50

- Reactive and perfect algorithms run on the order of nanoseconds, and are about 40 times faster than the anticipatory algorithm with depth = 2

ECSA, Sept. 2007

© Garlan 2007

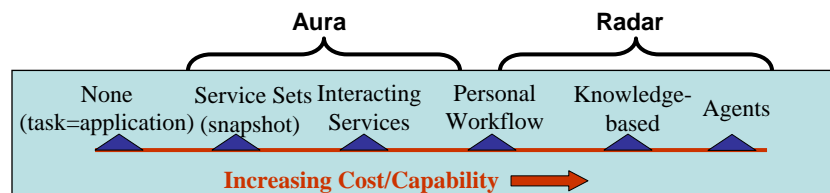
48

Common Architectural Themes and Lessons Learned

1. What is a task
2. Architectural drivers
3. Making it useful
4. Preventing bad things from happening
5. Gathering and exploiting context information
6. Compositionality and componentization

1. What is a Task

- Many possible answers
 - A single application or service
 - A collection of coordinated services
 - A workflow or process
 - A set of goals and constraints



Architecture Issues

- Nature of task has strong impact on architecture
 - Degree to which task management “understands” a task
 - Granularity of task assistance components
- Architecture challenges
 - Supporting a mixture of task types in a uniform way
 - Making it easy for a user to define new tasks
 - by example, by template, drag-and-drop
 - Handling multi-user tasks
 - Handling tasks in which users provide essential services

2. Architectural Drivers

- Task-oriented computing must address at least the following three drivers
 - **Compatibility:** dovetail with existing infrastructure, applications, interfaces
 - Often the most time-consuming aspect of engineering
 - **Extensibility:** must allow one to add new capabilities (tasks, environment monitors, policies, etc.)
 - **Adaptability:** system must change dynamically through learning, reconfiguration, etc.

3. Making it useful

- It is easy to get it wrong
 - Example: The Microsoft Paperclip
 - Example: Default folder for “Save a:
- Important tradeoff: proactivity versus reactivity
 - When do you automatically do something?
 - How do you involve the user?
 - How often do you consider reconfiguration?
 - How to support different degrees of trust in the system?

I think I know what is best for you, I am going to keep it enabled.



Architecture Issues

- Architectural mechanisms that can help
 - Interfaces to support transparency
 - what is the state of the system?
 - what is it automating? what does it believe?
 - Permitting mixed-mode computing
 - carrying out tasks is fundamentally a cooperative effort
 - user must be able to tell system how much automation they want on a particular task
 - Incorporating learning
 - so system gets better over time
 - Empirical evaluation
 - costly but worth it for reusable task support

3. Preventing bad things from happening

- Adaptability is at the heart of good task-oriented systems
 - examples: RADAR learning; Aura reconfiguration
- How can we make sure emergent behavior is correct or even reasonable?
 - example: components using machine learning may be different today than they were yesterday
 - example: Aura predictions may be based on incomplete knowledge

4. Gathering and exploiting context information

- Ability to effectively carry out task depends on having good context information
 - about the platform and computing infrastructure
 - about the user's state: location, state-of-mind,
 - about other people and situations
- Software architecture must build in mechanisms for context as first-class entities
 - These must be available for each part of the system involved in adaptation and reconfiguration.

Architectural Challenges

- Combining multiple sources of information
 - Example: user location
- Limiting scope
 - How much context is enough?
- Control
 - How is context information communicated and updated?
- Security and privacy
 - Can we make it context-dependent?

5. Compositionality and Componentization

- How can we create a component-based architecture for
 - resource monitors and predictors
 - learned expertise
 - task specialists
 - context information providers

Conclusion

- Task-oriented computing is necessary evolution of today's systems
- Requires new layers of system structure
 - To represent and carry out user-oriented tasks
 - To reduce user distraction and make users more effective
- Good software architectures for task-oriented computing are critical
 - Compatibility, Extensibility, Adaptability are critical architectural drivers
- Many interesting problems remain to be solved

The End
