Agent Languages

Overview

Requirements

Java

Tcl/Tk

Telescript

Evaluation

Requirements

for agent Languages

distributed programming

large-scale (tens of thousands of computers)

mobility

platform independence

security

distribution

automated installation and maintenance

inventory of installed software

impossible / not economical to do

manually

usage

fair pricing scheme through usage metering

user support

semi-automated, distributed / remote

cooperation

component-based software, CORBA, OLE, ActiveX, etc.

Example

Network Management Agents

SNMP Simple Network Management Protocol

standardized application-level IP protocol

SNMP console

central program used by the network administrator

graphical display of the network status[-2ex]

- network configuration (nodes, links)
- red icons for malfunctioning nodes
- performance monitoring

SNMP agent

server on every computing device on the network

- computer
- printer
- router
- modem

data collection agent for the console

SMNP Agents

automated network management

purpose

- network autonomy: networks should run themselves
- reliability: critical business function
- network performance optimization (routing, timing)
- early warnings for problems
- fault tolerance

implementation

simple server program in each device
connected to the network
remotely deployed and controlled

limitations

computation power in some devices

bandwidth

security

legacy networks

SMNP Agents

PAGE description

percepts

- messages from the network
- effects of the agent's activities
- external actions (reset, power)

actions

- receive, check, decode, convert messages
- compose, encode, check, send messages
- accept instructions from network management
- check internal status (self-check)
- collect and evaluate statistical information

- evaluate performance
- rerouting of message traffic

goals

- fault tolerance
- performance optimization
- agent autonomy
- performance optimization
- early warnings for problems

environment

computer network (LAN, Intranet)

mediator between the network and computing nodes

communication with other SNMP agents

Java

programming for the Web

origin

software development for consumer electronics
extension / simplification of C++

properties

- platform-independent (hardware, operating system)
  compiled into a portable binary format (bytecode)
- multi-threaded
- interactive
- safe to transfer over networks (viruses)
- secure (access to private resources limited)

- object-oriented

Objects

and Java

encapsulation

implementation details are hidden

reusability

structured programs that can be reused as building blocks

polymorphism

operations are adapted to the objects they are used on

messages

transfer of information between objects

Java Libraries

collections of basic routines

`java.lang`

basic types, fundamental classes

Object, Class, threads, exceptions, wrappers

`java.io`

input/output functions

streams, random-access files

`java.net`

network functions

sockets, URLs, telnet, protocols

`java.util`

container and utility classes

Dictionary, HashTable, Stack, encoding and decoding for date and time classes

`java.awt` Abstract Windowing Toolkit

abstract layer for user interface design

designed for an evolving environment

Java Environment

execution of programs

Java interpreter

executes Java bytecodes directly

Java compiler

produces instruction for the Java virtual machine

some instructions are not allowed in the bytecode

Java virtual machine

platform-independent runtime environment translates the bytecode into the language of the underlying hardware

just-in-time-compilation (at execution time)

bytecode verifier

checks legality of code

assumes that no bytecode is sure bytecode that violates language constraints is not executed

authentication and security must be balanced with performance

Applets and Applications

Java-based programs

Java applet

Java programs for Web browsers

no reading and writing of files in the client file system

transferable via network

platform-independent

Java aplication

regular program without restrictions

Java security

applet security manager enforces applet restriction

only one security manager per browser, can't be replaced, overwritten, or altered

Tcl/Tk

agent toolbox

origins

general purpose scripting language for tool development

- Tcl (Tool Command Language)
- Tk (Tool Kit) extension of Tcl for the creation of graphical user interfaces

usage

development of applications with sophisticated user interfaces
often used for agent-oriented systems

properties of Tcl

- simple language
- extensible with user-defined constructs
- versatile for inclusion in new tools

important concepts in Tcl

- *string* as single data type: everything is a string
- quotation mechanism
- a *command* is a word followed by a list of words that act as arguments
- *control structures* can be extended and added
-

toolkit extension for Tcl

features of Tk

- widgets for text, images, drawings
- geometry manager
- binding mechanism to assign actions to user events
- option database to control behavior of Tk components

usage

graphical user interface development

concise

easy to use

considerable reduction in development time (10-fold) over C++/Motif

Safe Tcl

safe and unsafe Tcl commands

padded cell security

dual set of interpreters

one is trusted and unrestricted, runs in kernel space

the other untrusted and restricted, runs in user space

trusted commands similar to system calls in OS

provided by the trusted interpreter to the untrusted one

allows specific actions for guest agents while still maintaining overall control

unsafe commands (examples)

general file access, exec for the invocation of other programs

limitations

resource management (CPU limits, memory space, disk space)

agent delivery mechanism is open and extensible

control of applications is platform-dependent to a large degree

easier to handle than the "sea of objects" security model (Java, Telescript)

Telescript

commercial platform for agents

origin

operating system for personal intelligent communicators (Magic Cap)

General Magic (`http://www.genmagic.com/`) spinoff from Apple

purpose

development tool for mobile agents

active networks for locating distributed information

features

- language
- engine
- protocol

- security regime

Telescript Use

remote programming for agents

remote operation

agents carrying data and instructions are sent over the network

Telescript agents

active entities behaving intelligently

encapsulate the instructions of users together with data and permits

permits

capabilities granted and limited by authorities (users, hosts)

travel

movement between locations to services offered remotely

achieved by the go command

meeting

interaction between agents in the same location

exchange of information, negotiations of transactions

Telescript places

stationary locations to be inhabited by local and outside agents

Telescript engines

collection of Telescript places

Telescript clouds

collection of Telescript engines

provide support services (registration, directory assistance)

Telescript Language

technical issues

objects

object-oriented language, classes, inheritance

binding and linking

dynamic, to allow the utilization of services at remote locations

execution

via interpreters in engines

portability

virtual machine for machine-independence

persistence

nonvolatile memory is used to protect agains computer failure

engines write to disk periodically in a transparent way

Telescript Engines

purpose

accomodate agents and places

provide services via APIs (Application Programming Interface)

enable transportation of agents

Storage API

provide access to permanent storage

used for persistence

Transport API

access to communication facilities for transporting agents

External API

interaction with other applications

potential security risk since the security layer is bypassed

Telescript Security

identification

every agent and place has a unique identity

credentials

agents must have permits for places and activities

encryption

is used to transfer agents betwen engines

interpretation

to prevent access to critical resources

transportation

single methos `go` to support movement of agents

Evaluation

of agent languages

safety

the host computer and applications are safe from bugs and crashes of a hosted agent

agent vs. virus: different only in the intent of the author

security

the actions of an agent are restricted

access to data and resources only with permission

pointers are a security risk

portability

platform-independence (hardware and operating system

dynamic binding (at executino time) is

important for agents

performance

interpreted vs. compiled

reuse

components can be combined into applications

mobility

programs are sent over the net and executed remotely

interpreted languages usually are more appropriate than compiled languages

Agent Languages

Summary

Requirements

safety, security, portability, mobility, reuse, performance

Java

object-oriented, dynamic, clean, portable,secure

Tcl/Tk

toolset for agent development, extensions for user interface implementation, safety

Telescript

object-oriented, dynamic, interpreted, network programming language, security schemes, single abstraction for agent transportation (go

Evaluation

of agent languages