

Multiagentové systémy

Dr. Andrej Lúčný

KAI FMFI UK

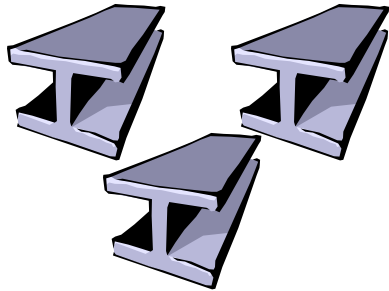
andy@microstep-mis.com

<http://www.microstep-mis.sk/~andy>

Opakovanie

- Aké rozdiely sú medzi neštruktúrovaným, štruktúrovaným, objektovo-orientovaným a agentovo orientovaným programovaním ?
 - ako definujeme pojem agent ?
-
- dnes: základné pojmy MAS +
nepovinná prerekvizita: VRML

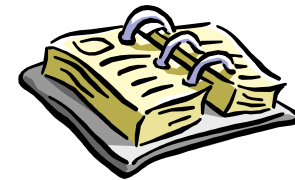
Multiagentový systém



Agenty



*Komunikácia
medzi
agentami*

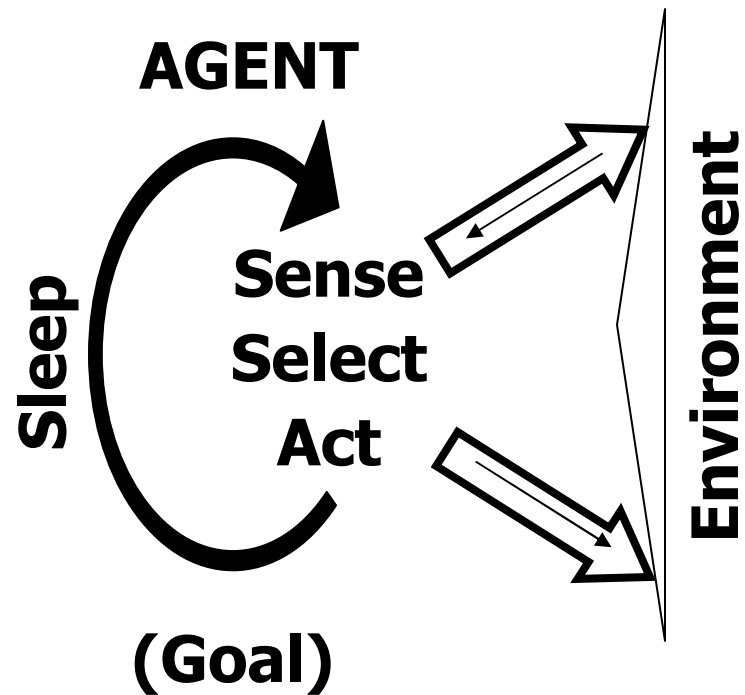


*Metóda
tvorby*

ARCHITEKTÚRA

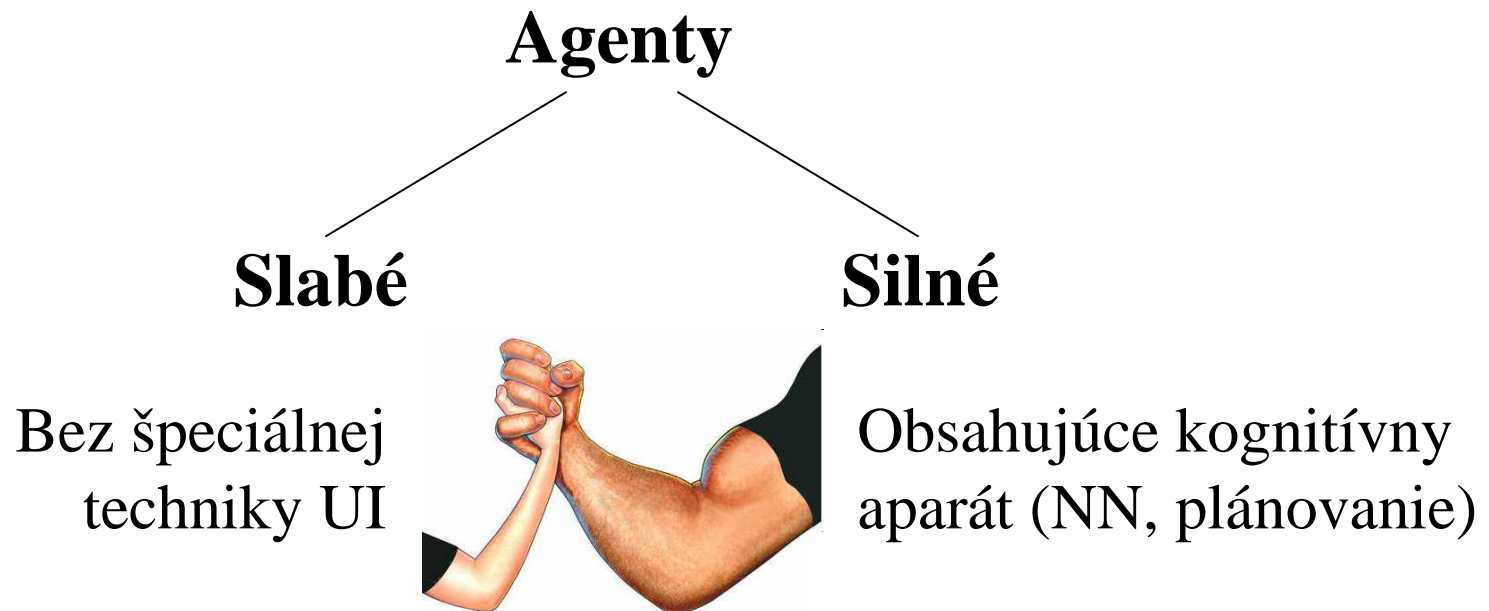
Agenty

Agent - proces



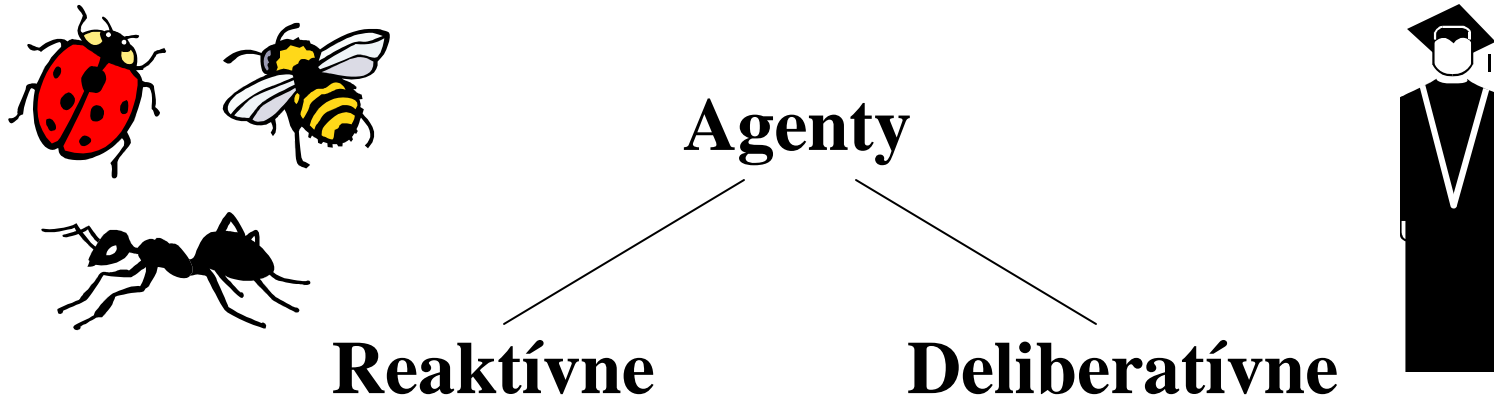
Agent je proces, ktorý neustále (opakovane) vníma svoje prostredie a na základe toho volí a vykonáva v ňom akcie, sledujúc určitý cieľ

Agent – klasifikácia podľa prítomnosti UI mechanizmu



[Nick Jennings, Michael Wooldridge]

Agent – klasifikácia podľa voľby



Reagujú = volím toto,
lebo sa to za daných
okolností robieva

Rozhodujú sa = volím
toto, lebo z odhadnutých
následkov možných
akcií, sa táto najlepšie
pozdáva

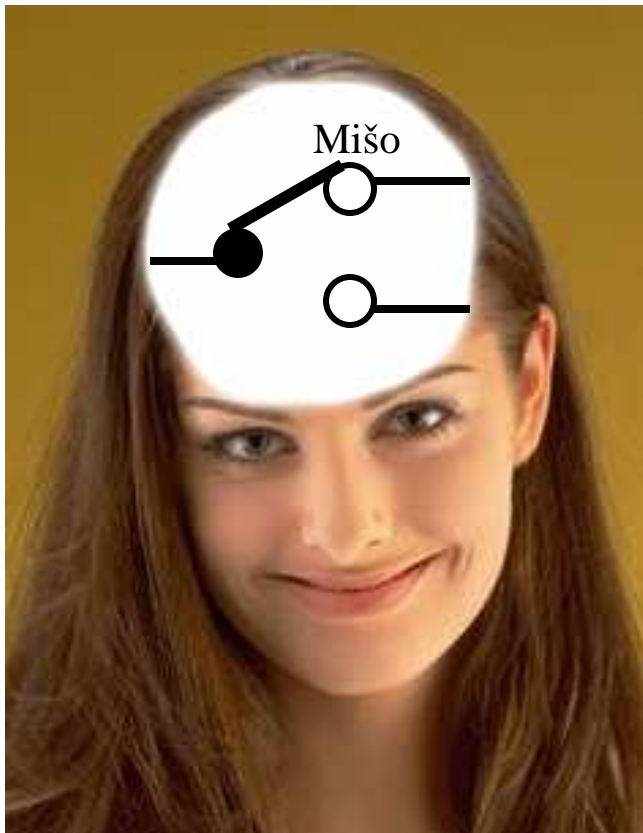
Agent – klasifikácia podľa voľby



Deliberatívny agent

- potrebuje model sveta
- spravidla používa jednotný reprezentačný jazyk
- voľba je spravidla realizovaná vzatím prvej akcie najhodnotnejšieho vygenerovaného (čiastočného) plánu

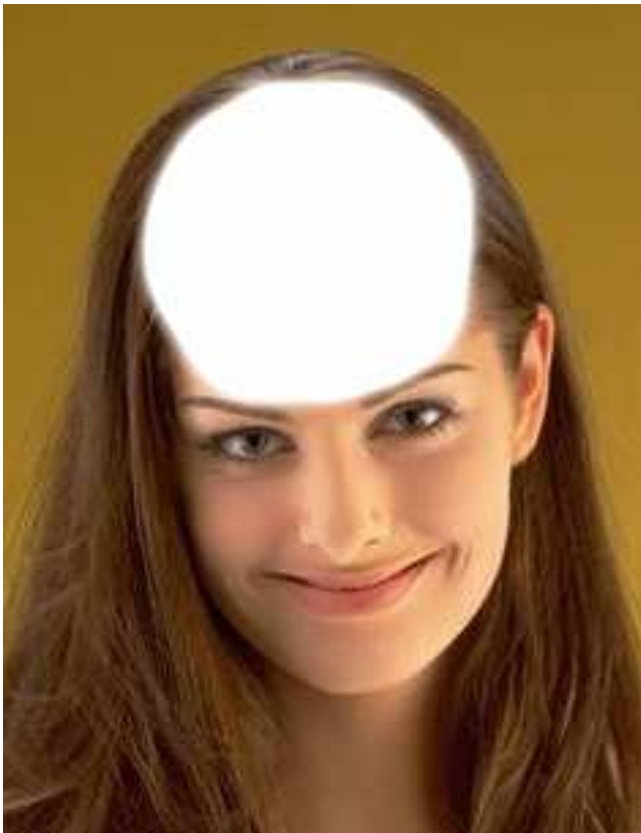
Agent – klasifikácia podľa voľby



Reaktívny agent

- nepotrebuje model sveta
- voľba je spravidla realizovaná rozhodovacím stromom alebo KSA, ktoré volajú výpočtové procedúry

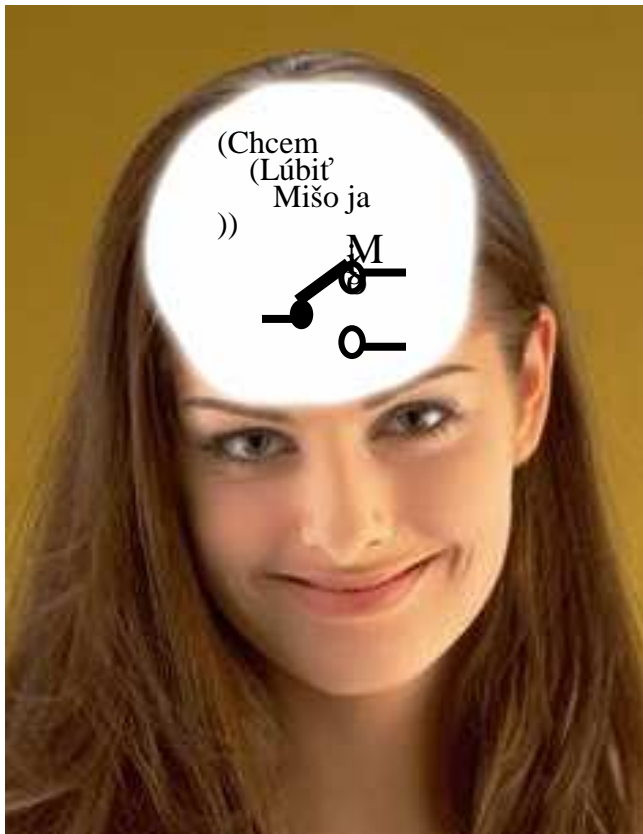
Agent – klasifikácia podľa voľby



Čisto reaktívny agent

- nemá vnútorný stav
- Je odkázaný na informácie vo svojom prostredí, tieto môže aktívne vytvárať pre svoju potrebu (uzlík na vreckovke)

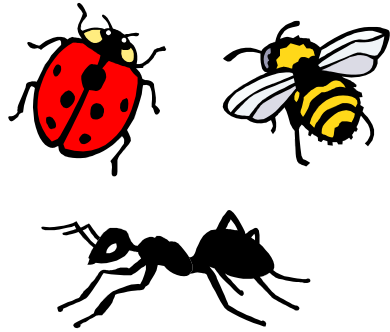
Agent – klasifikácia podľa voľby



Hybridný agent

- kombinácia rôznych techník
- Spravidla reaktívne rieši to čo treba riešiť rýchlo, deliberatívne to na čo máme dostatok času
- deliberatívnymi postupmi upravujeme činnosť reaktívnych zložiek

Agent – Klasifikácia podľa cieľa



Cieľ



Implicitný

Explicitný

Typický pre reaktívne
agenty

Je zakódovaný do
voľby akcie a v
explicitnej podobe je
iba v plánoch
programátora

Je vyjadrený v
reprezentačnom
jazyku v pamäti
agenta

Typický pre
deliberatívne agenty

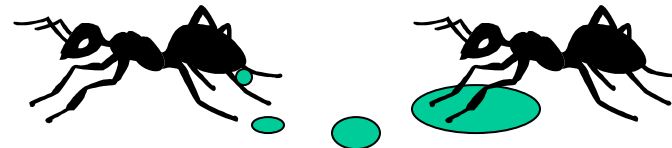
Komunikácia medzi agentami

Komunikácia medzi agentami

Komunikácia

priama - adresná

nepriama - cez prostredie



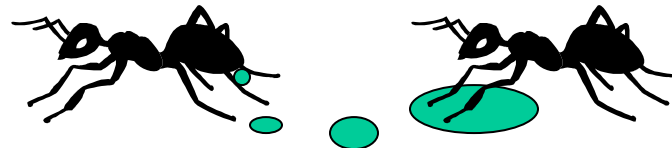
Komunikácia medzi agentami

Referencia protistrany

pevná adresa



pomenované odkazy v pevne adresovanom prostredí



Komunikácia medzi agentami

Komunikované dáta:

- **holé dáta** 00001010 0000000
- **dáta s definovaným typom** Integer 10
- **reprezentačný jazyk** “(vek 10 rokov)”
(štruktúrované, semištruktúrované) “<vek> ten </vek>”

Komunikačná obálka:

- **komunikačný jazyk** “(ask-if (...))”

Reprezentačný jazyk

- KIF - jednoznačne určuje ako sa niečo povie (definuje jedinú sémantiku slovníka)
- XML – veľa rôznych výpovedí môže znamenať to isté, ale syntax umožňuje preložiť jednu formu na druhú

KIF

- dohodnutá syntax – LISP
- dohodnutá sémantika – obrovský slovník

Keď chce programátor zapísať bod [1, 1] v rovine,
pozrie do slovníka a zistí že je to

(point (euclid-coordinates 1 1))

alebo

(point (polar-coordinates 1.1412136 0.7853981))

KIF

(instance simultaneousProcess BinaryPredicate)

(documentation simultaneousProcess

"(simultaneousProcess ?Proc1 ?Proc2) means that processes ?Proc1 ?Proc2 cooccur at the same object, but not necessarily at the same region. "

)

(<=>

(simultaneousProcess ?Proc1 ?Proc2)

(and (cooccur ?Proc1 ?Proc2)

(exists (?object)

(and (patient ?Proc1 ?Object)

(patient ?Proc2 ?Object))))))

XML

```
<tag attribute=value ... >  
    data  
</tag>
```

tag s dátami

```
<tag attribute=value ... />
```

prázdny tag

```
<? ... ?>
```

direktíva

```
<!-- remark -->
```

poznámka

```
<![CDATA[ ... ]]>
```

raw data

XML

```
<?xml version="1.0" ?>
```

```
<announcement>
```

```
  <going-to who="007">
```

```
    10 10 <stop/>
```

```
  </going-to>
```

```
</announcement>
```

```
<?xml version="1.0" ?>
```

```
<oznam-idem-na>
```

```
  007 10 10 <stop/>
```

```
</oznam-idem-na>
```

XML Transformer XSLT

Elementy:

<xsl:attribute-set>
<xsl:decimal-format>
<xsl:import>
<xsl:include>
<xsl:key>
<xsl:namespace-alias>
<xsl:output>
<xsl:param>
<xsl:preserve-space>
<xsl:strip-space>
<xsl:template>
<xsl:variable>
<xsl:script>

match= name= priority= mode=

Instrukcie template:

<xsl:apply-imports>
<xsl:apply-templates>
<xsl:attribute>
<xsl:call-template>
<xsl:choose>
<xsl:comment>
<xsl:copy>
<xsl:copy-of>
<xsl:element>
<xsl:fallback>
<xsl:for-each>
<xsl:if>
<xsl:message>
<xsl:number>
<xsl:processing-instruction>
<xsl:text>
<xsl:value-of>
<xsl:variable>

select= disable-output-escaping=

XSLT

```
<?xml version="1.0" ?>
```

```
<xsl:stylesheet version="1.0" xmlns:xsl= http://www. w3.org/1999/XSL/Transform>
```

```
<xsl:template match="/annoucement">
```

```
    <oznam-idem-na>
```

```
    <xsl:apply-templates select="going-to"/>
```

```
    <xsl:value-of select="going-to"/>
```

```
    </oznam-idem-na>
```

```
</xsl:template>
```

```
<xsl:template match="going-to">
```

```
    <xsl:value-of select="@who"/>
```

```
</xsl:template>
```

XSLT

<?xml version="1.0" ?>

<announcement>

<going-to who="007">

10 10 <stop/>

</going-to>

</announcement>



<?xml version="1.0" ?>

<oznam-idem-na>

007 10 10 <stop/>

</oznam-idem-na>

Komunikačný jazyk

- rečové akty – KQML
- syntax – LISPOVSKÉ RÁMCE

(performatív

:parameter hodnota

:parameter hodnota

)

KQML

Performatívy

Achieve	sender žiada receivera o vykonanie contentu
Advertise	sender ponuka službu a tvar žiadosti v contente
Ask-all	sender žiada o všetky odpovede na otázku v contente
Ask-one	sender žiada o jedinú odpoveď na otázku v contente
Broker-one	sender sa pýta kto vie zodpovedať otázku v contente
Delete-one	sender oznamuje že content už neplatí – zmažte si ho
Insert	sender oznamuje že content platí – zapamätajte si
Ready	sender oznamuje že už vie urobiť content
Register	sender oznamuje že existuje
Recommend-one	sender žiada aby mu odporučili agenta ktorý pozná content
Recruit-one	sender žiada aby mu našli agenta ktorý pozná content
Sorry	sender nemá k dispozícii požadovanú informáciu
Standby	sender oznamuje že vie urobiť content
Subscribe	sender žiada aby mu dali vedieť keď sa zmení content
Tell	sender posiela receiverovi čo sa ho pýtal

...

KQML

Parametre

:sender

:receiver

:from

:to

:in-reply-to

:reply-with

:language

:ontology

:content

...

kto posiela správu

kto má prijať správu

od koho správa pochádza

komu je správa určená

žiadosť o označenie odpovede

označenie odpovede

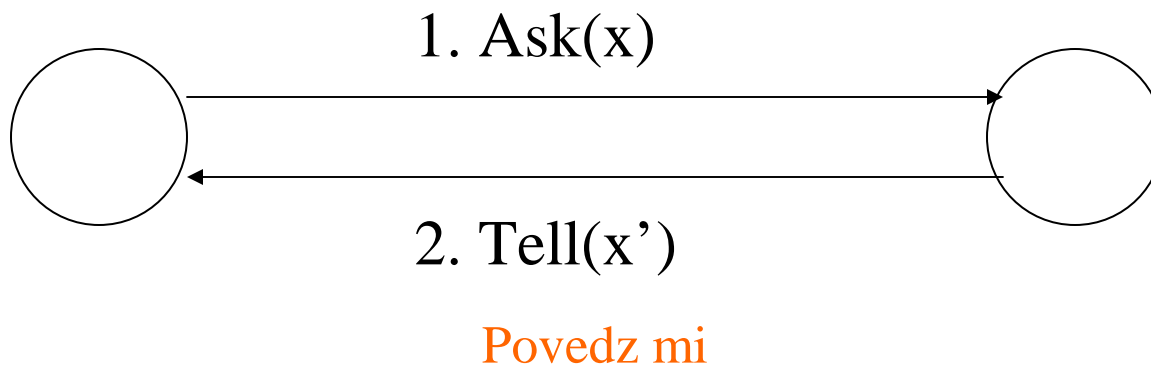
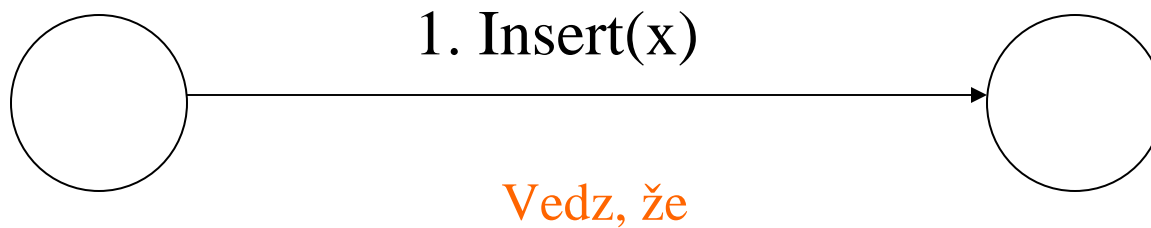
názov syntaxe contentu

názov sémantiky contentu

obsah správy

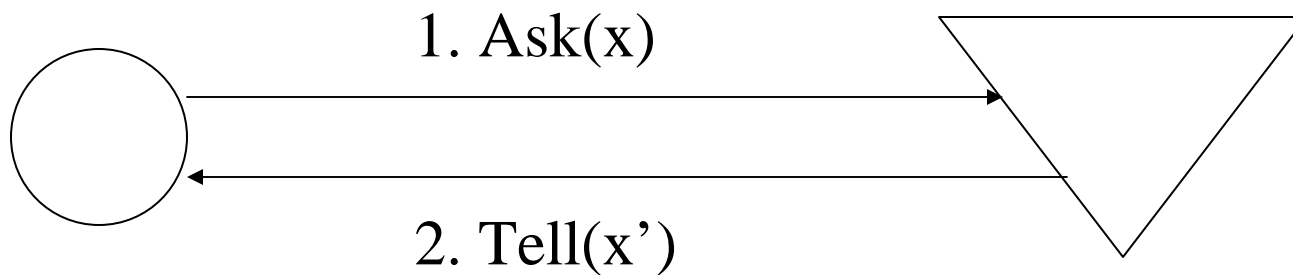
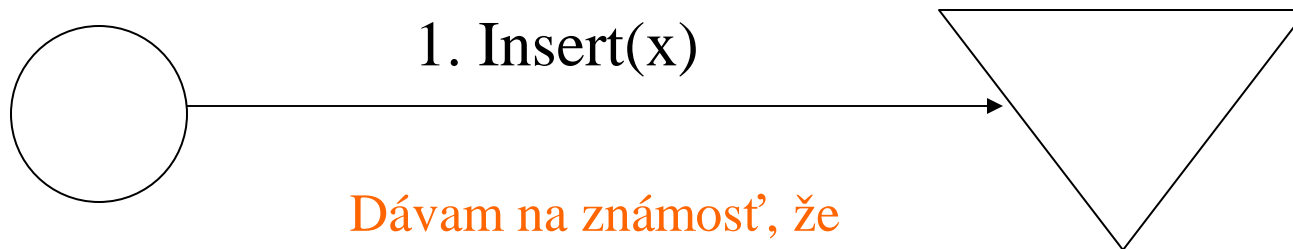
KQML- scenáre

Priama komunikácia



KQML- scenáre

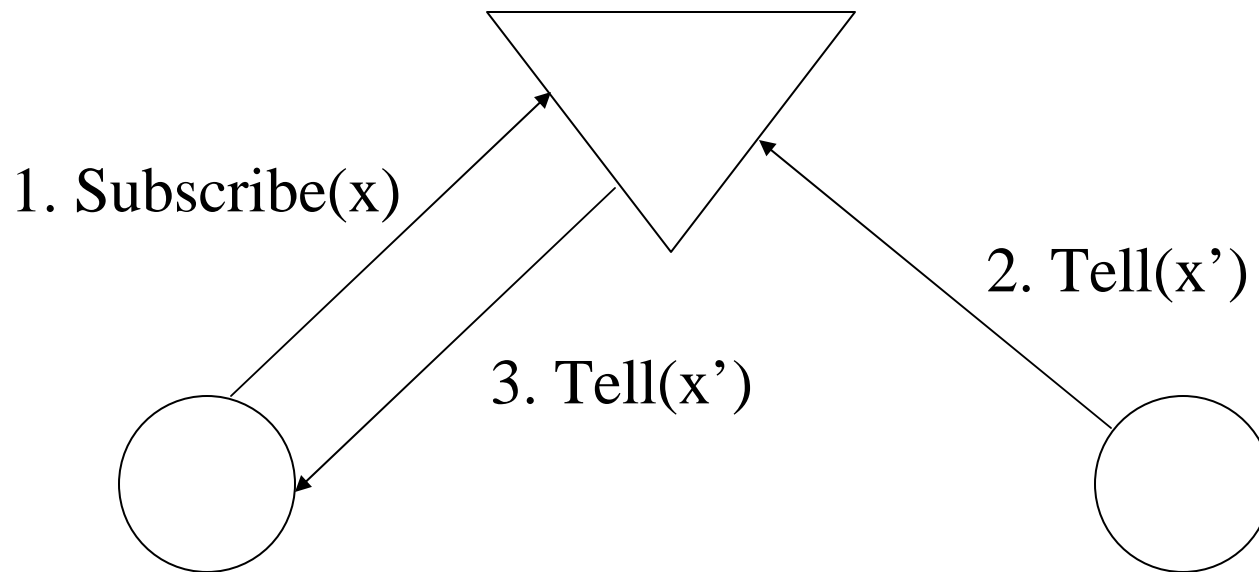
Nepriama komunikácia



Čo sa dalo naposledy na známosť o tomto ?

KQML- scenáre

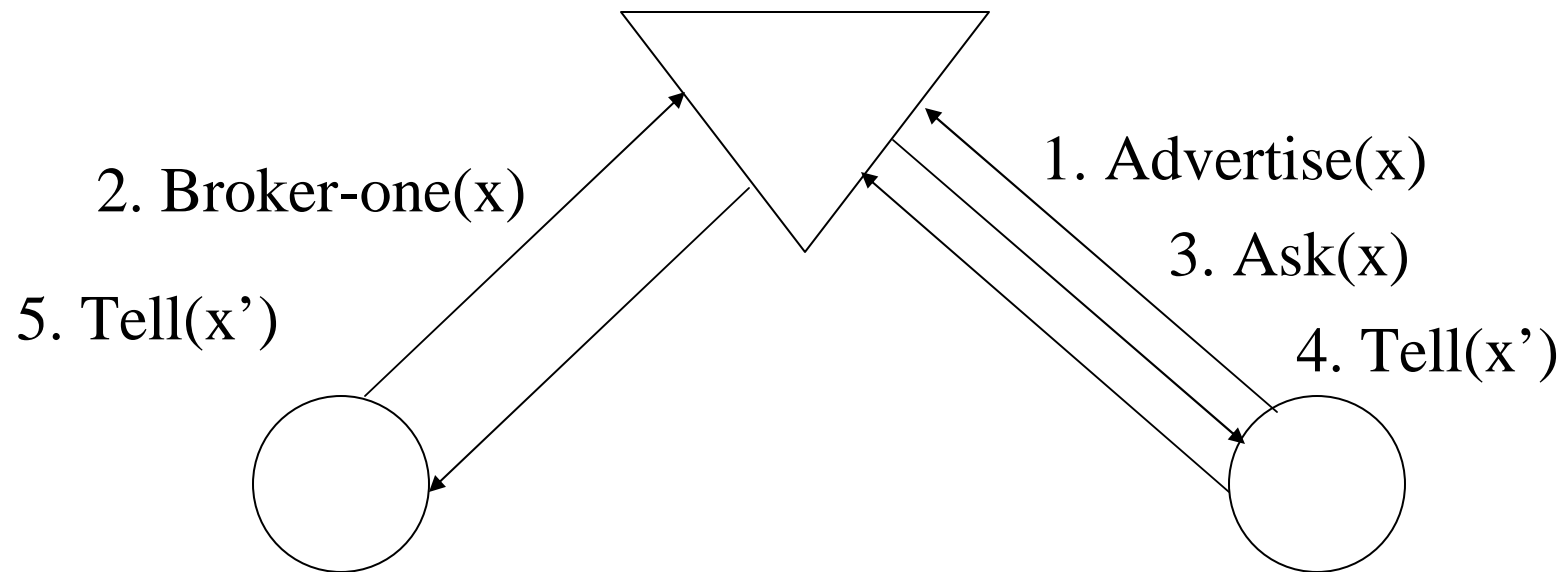
nepriama komunikácia – request on trigger



Ked' to niekto povie chcem to tiež vediet'

KQML- scenáre

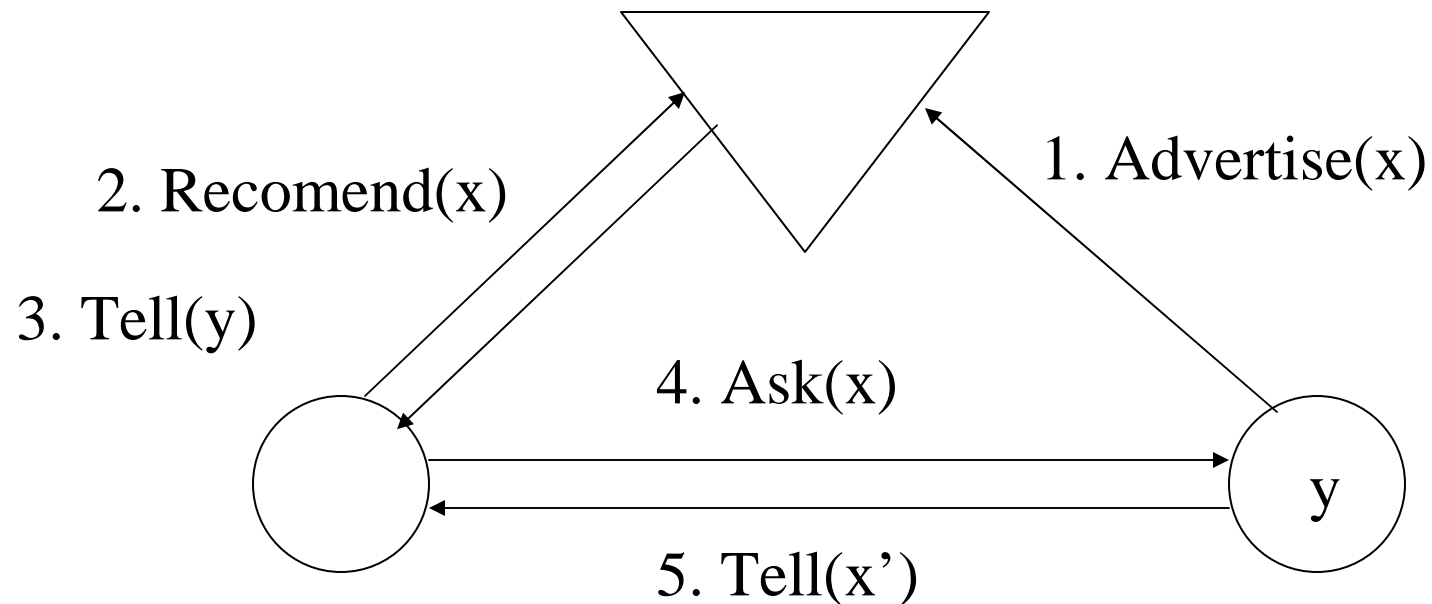
nepriama komunikácia – request bidding



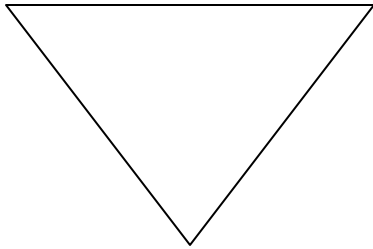
Zistite mi odpoved'

KQML- scenáre

priama komunikácia – request bidding



Povedzte mi kto vie odpovedať

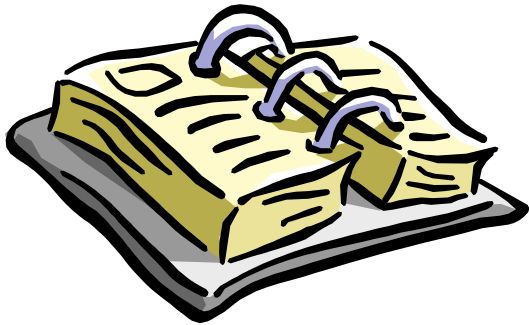


čo je to ?

- meta-agent
- facilitator
- container - mediator
- space
- (environment)

Metóda tvorby

Metóda tvorby



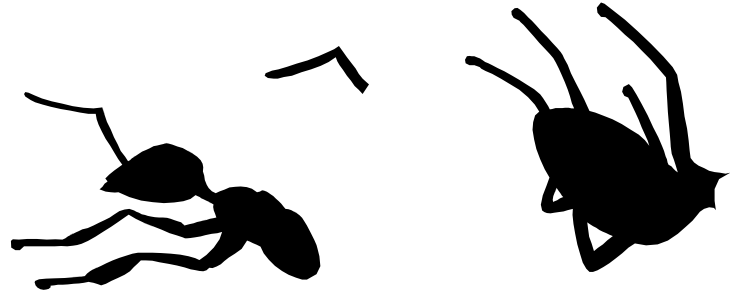
Systemy založené na:

- **reaktívnych agentoch**
- **deliberatívnych agentoch**
- **hybridných agentoch**
- **hybridné systémy**

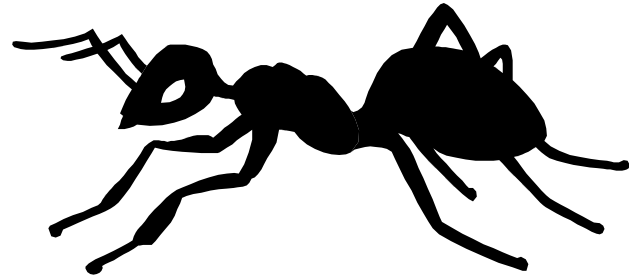
**Ako zabezpečiť aby sa nám podarilo vytvoriť
systém, ktorý vytvoriť zamýšľame ?**

Trojvrstvá architektúra

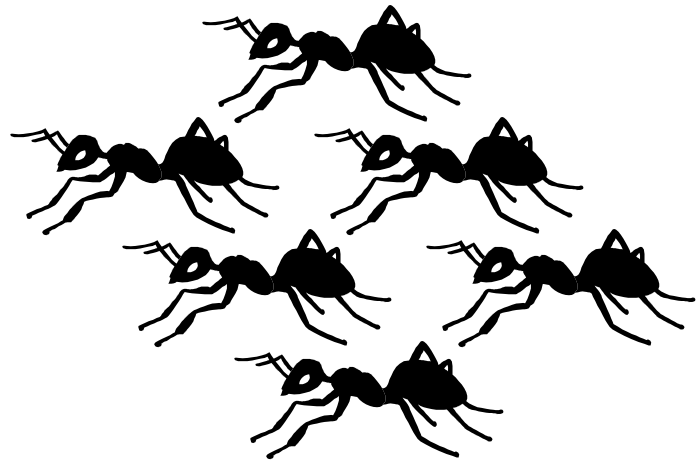
- moduly



- agenti

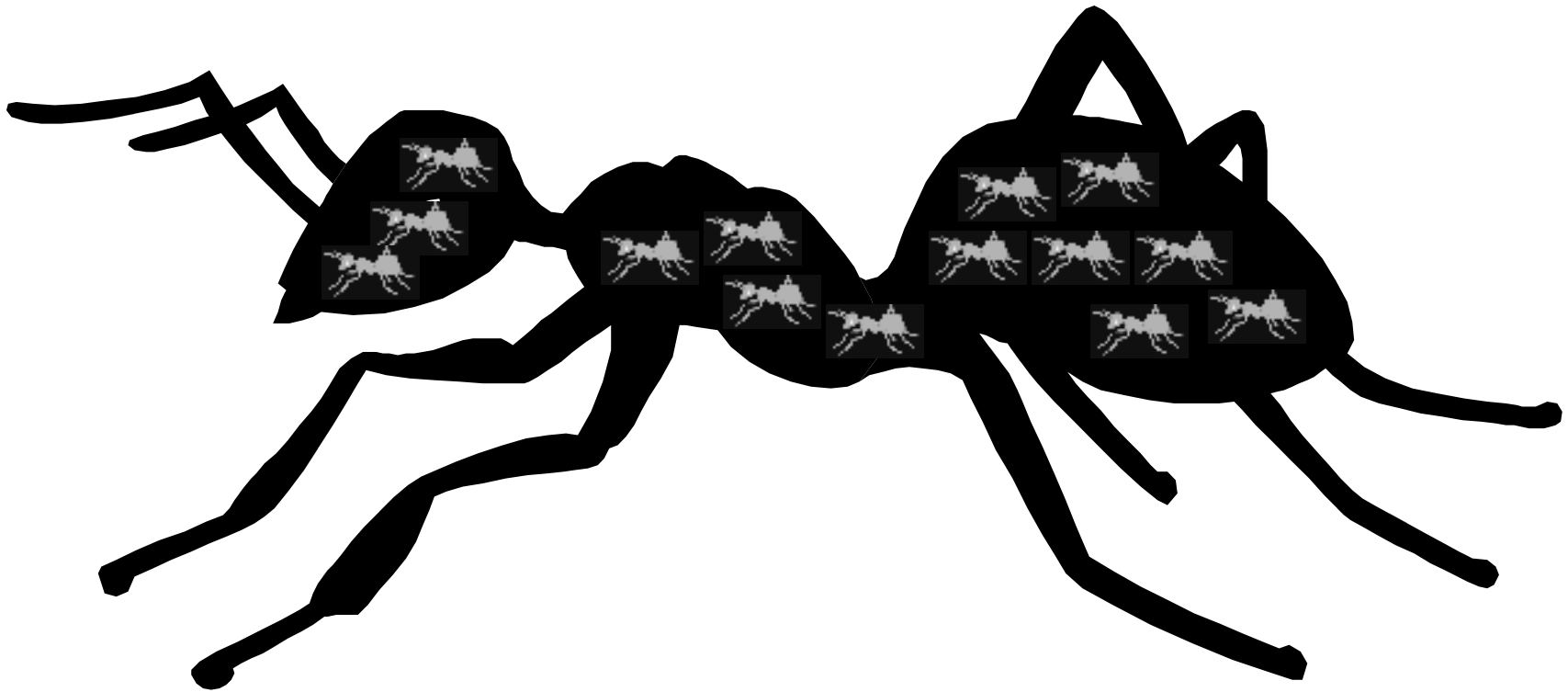


- kolónia agentov



bežná pre simulačné prostredia

Hierarchická architektúra



Agent môže byť potenciálne
multiagentový systém nižšej úrovne

bežné pre AOP