AMETAS

Hinweise zur Erstellung von Anwendungen finden sich im **UsersGuide**

http://www.ametas.de/en/docs/UsersGuide/

Beispiele sind in den Kapiteln
- Implementing agents
- Messages and communication

zu finden.

Grundlegende Informationen.

1. Jeder Agent muss eine Methode *invoke* implementieren. Wenn der Agent diese Methode wieder verlässt, dann wird er automatisch beendet.

2. Ein Agent kann nur mittels Nachrichten kommunizieren, d.h. er selbst kann sich dem Anwender nicht erkenntlich machen (etwa durch ein Fenster oder Ausgabe in ein Terminal).

3. Ein Agent kann in die Logdatei der Stelle schreiben:

```
...
m_Driver.output("Dies ist eine Ausgabe.");
...
```

Da alle Stellennutzer dort hineinschreiben sowie auch alle anderen Komponenten, ist das Stellenlog häufig sehr voll, und Ausgaben können übersehen werden.

4. Um einen anderen Agenten oder allgemein Stellennutzer zu finden, dient die Mediation.

```
...
AMETASMediationRequest mrq = new AMETASMediationRequest("TestService", true);
AMETASMediationResult[] amrs = m_Driver.request(mrq);

if (mrq==null) m_Driver.output("Ich habe einen Fehler in der Abfrage");
else {
  if (mrq.length==0) m_Driver.output("Ich habe keinen geeigneten Stellennutzer gefunden.");
  else {
    puid = amrs[0].getPlaceUserID();
    m_Driver.output("Ich habe mindestens einen geeigneten Stellennutzer gefunden.");
}
...
```

5. Um eine Nachricht einem anderen Stellennutzer zu schicken, braucht er dessen Adresse. Angenommen, diese liegt in puid, dann sieht das so aus:

```
...
Object[] aBody = new Object[3];
as[0] = "Hallo";
as[1] = "Antwort";
as[2] = new Integer(42);

AMETASMessage mes = new AMETASMessage(puid, as);
int nStatus = m_Driver.depositMessage(mes);
if (nStatus==AMETASErrors.OK) {
  m_Driver.output("Alles OK");
}
else {
  m_Driver.output("Mist");
}
...
```

6. Um zu migrieren, ruft der Agent eine Methode go auf dem Treiber auf.

```
...
try {
  m_Driver.go("andere.stelle.andere.domaene.");
}
catch (MigrationAbortedException e) { ... }
catch (NoActiveDestinationException e) { ... }
catch (UnknownDestinationException e) { ... }
catch (RejectedException e) { ... }
catch (AccountExpiredException e) { ... }
...
```

Die Exceptions sind selbsterklärend (siehe sonst auch die API-Beschreibung). Die ersten vier sind Subklassen von *MigrationException*. *AccountExpiredException* ist eine RuntimeException und muss nicht unbedingt gefangen werden.

7. Ein Agent kann leicht gestartet werden, aber das Stoppen ist wesentlich schwieriger. Deshalb gibt es nur ein "startapp"-Skript, aber kein "stopapp". Im Zweifelsfall einfach die Stelle komplett stoppen.

8. Bei Benutzeradaptern wird gerne der Fehler gemacht, den Treiberthread in invoke nicht "festzuhalten". Wenn der Thread die invoke-Methode verlässt, wird der Stellennutzer beendet.

```
...
m_Driver.idle();
...
```

hält den Thread fest, bis jemand ein m_Driver.wakeup() aufruft. Dies kann etwa der AWT-EventThread tun, wenn man auf einen Knopf ("Beenden") drückt oder das Fenster schließt (WindowListener).

9. Ich empfehle, die beiden Anwendungen "MeinAgent" und "MeinBenutzeradapter" zu studieren und zu testen.

10. Windows-Rechner verfügen nicht immer über eine vollständige DNS-Konfiguration. Dies äußert sich darin, dass in der Abfrage zum Rechnername die Windows-Domäne/Arbeitsgruppe angezeigt wird (z.B. "Rechner.WORKGROUP"). In diesem Fall muss man die IP-Adresse angeben (ipconfig /all).

11. Rechner hinter einer NAT-Firewall (welche das innere Netz maskiert) können keine Agenten von außerhalb des Intranets empfangen. Man kann dann entweder nur Agenten innerhalb des Intranets verwenden oder muss die Agentenstelle auf dem Firewallrechner installieren (wenn es sich nicht nur um eine einfache Hardwarebox handelt).

12. AMETAS benötigt für das Empfangen von Agenten einen von außen zugänglichen Port. Normalerweise werden Verbindungsanfragen von außen durch Firewalls abgeblockt. Es ist dafür zu sorgen, dass dieser Port von außen erreicht wird. Die Firewall muss entsprechend konfiguriert werden.