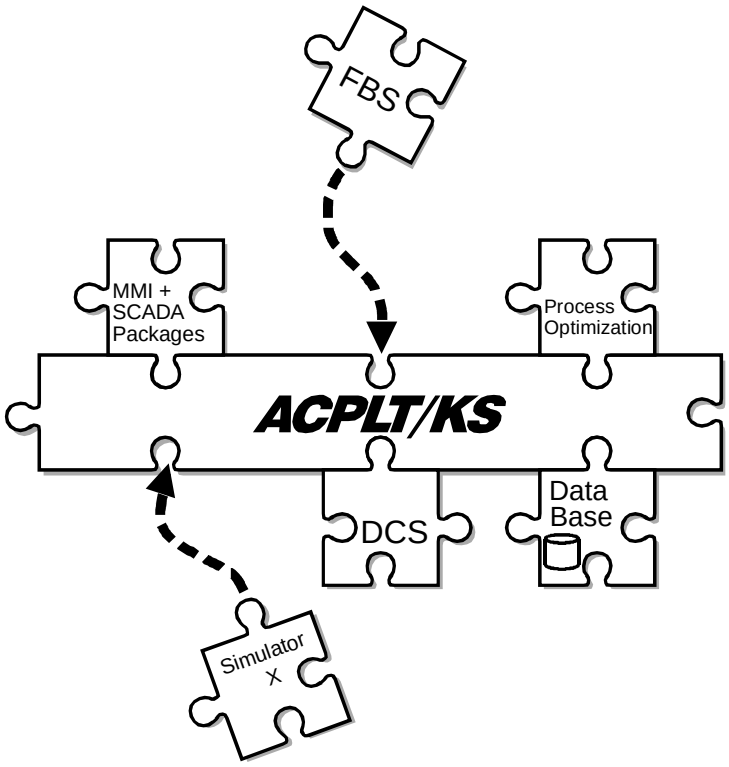
ACPLT/KS

Technologiepapier Nr. 3:
Manager, Server und Klienten



Inhalt

Technologiepapier Nr. 3 – Ausgabe 04.97.1
ersetzt: Ausgabe 01.97.1

Ansprechpartner: H. Albrecht
Lehrstuhl für Prozeßleittechnik der RWTH Aachen
D-52056 Aachen
Tel.: +49 241 80-7703
Fax.: +49 241 8888-238
eMail: ks@plt.rwth-aachen.de
WWW: http://www.plt.rwth-aachen.de/ks

1 Einleitung

Das Kommunikationssystem ACPLT/KS (im folgenden auch „KS“ genannt) verbindet im Bereich von Anwendungen der Prozeß- und Betriebsführung rechnergestützte Werkzeuge untereinander und mit Prozeßleitsystemen. Dieses Technologiepapier beschreibt die drei Typen von Teilnehmern am KS-Protokoll: Manager, Server sowie Klienten (Abbildung 1.1). Weiterhin wird der Ablauf der Kommunikation zwischen diesen Teilnehmern definiert.

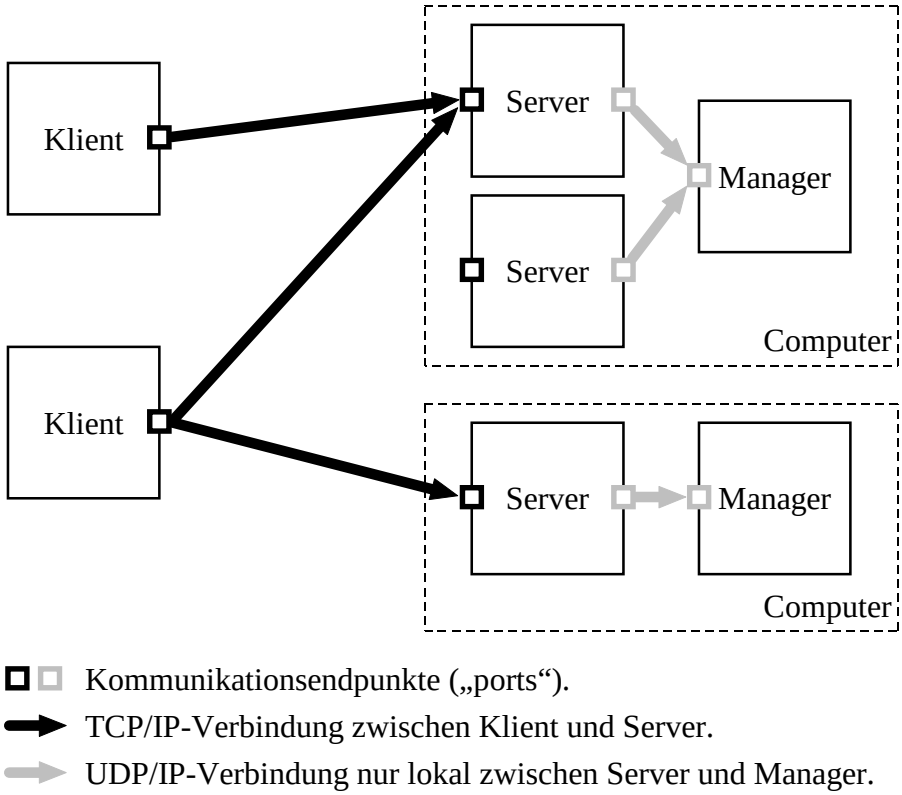


Abbildung 1.1: *ACPLT/KS-Klienten, -Server und -Manager.*

Jeder KS-Manager ist für die Verwaltung aller derjenigen KS-Server zuständig, die auf einem gemeinsamen Rechner gestartet wurden. Der Manager dient dabei als Informationsquelle, unter welchem Kommunikationsendpunkt („ports“) welcher KS-Server erreichbar ist [1]. Die KS-Server schließlich stellen Informationen bereit („exportieren“ Daten), die von KS-Klienten abgefragt werden können.

2 Der KS-Manager

Ein Ziel des Kommunikationssystemes ACPLT/KS ist, den Anwender soweit wie möglich von jeglichen Konfigurationsarbeiten zu entbinden. Um beispielsweise eine PLT-Stelle abzufragen, muß der Anwender neben dem Namen der PLT-Stelle nur noch wissen, wie der zugehörige KS-Server heißt und auf welcher Maschine dieser läuft.

Damit ein Klient nun eine RPC-Verbindung über TCP/IP (oder auch UDP/IP) zu dem Server aufbauen kann, müssen ihm die folgenden Informationen bekannt sein:

- IP-Rechneradresse des Servers, das verwendete Protokoll (TCP/UDP), Portnummer.
- RPC-Programmnummer, KS-Protokollversion (zur Zeit = 1).

Die IP-Rechneradresse ergibt sich aus dem Namen derjenigen Maschine, auf der der Server läuft. Für Verbindungen zwischen KS-Servern und KS-Klienten wird weiterhin nur TCP/IP eingesetzt, so daß damit das Internet-Protokoll ebenfalls bekannt ist.

Alle ACPLT/KS-Teilnehmer, egal ob Manager, Server oder Klienten, benutzen das gleiche KS-Protokoll, das auf ONC/RPC als „Transportmedium“ aufgesetzt ist (siehe auch [1]). Folglich teilen sich alle ACPLT/KS-Teilnehmer die gleiche einheitliche RPC-Schnittstelle, die innerhalb des Internets durch die weltweit eindeutige RPC-Programmnummer `0x49678` identifiziert wird. Die RPC-Programmnummern werden zentral von der Firma Sun verwaltet und auf Anfrage vergeben.

Jetzt fehlt noch die Portnummer, damit die über das Netzwerk verschickten Daten eindeutig einem Programm beziehungsweise Prozeß innerhalb der durch die IP-Rechneradresse spezifizierten Maschine zugeordnet werden können. Bei den Portnummern unterscheidet man zwischen dynamisch vergebene Portnummern sowie den sogenannten „well known port numbers“. Im letzteren Fall sind häufig benötigten Programmen feste, von vornherein bekannte, Portnummern zugeteilt.

RPC-Server benutzen im Allgemeinen (von wenigen Ausnahmen wie beispielsweise NFS abgesehen) dynamisch vergebene Portnummern. Damit ein RPC-Klient eine Verbindung zu einem derartigen RPC-Server aufbauen kann, existiert ein spezieller RPC-Server, den man als „Portmapper“ bezeichnet. Er stellt ein dynamisches Verzeichnis bereit, in dem diejenigen – zur Zeit auf einer Maschine aktiven – RPC-Server verzeichnet sind, die dynamisch vergebene Ports verwenden. Ein Klient erfragt zuerst beim Portmapper anhand der weltweit eindeutigen RPC-Programmnummer sowie einer zusätzlichen Versionsnummer die Portadresse des gewünschten RPC-Servers. Danach baut der Klient eine direkte Verbindung zum RPC-Server auf.

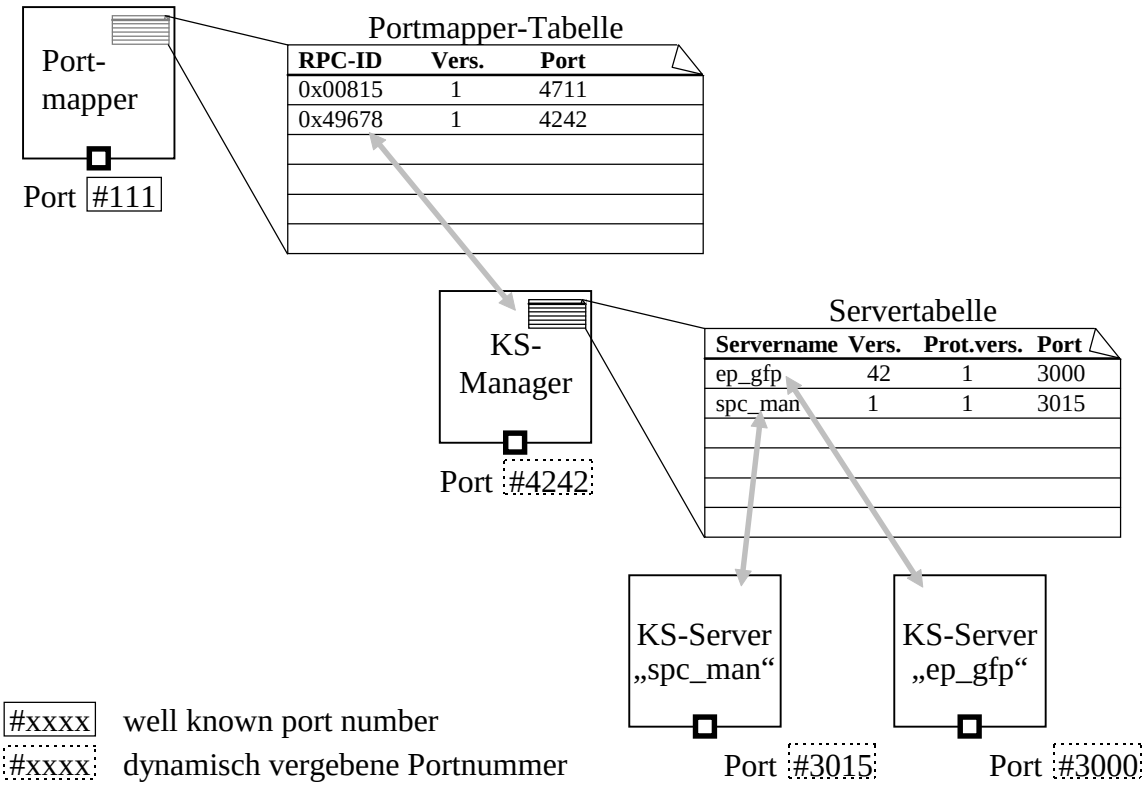


Abbildung 2.1: *Der KS-Manager verwaltet alle KS-Server eines einzelnen Rechners.*

Diese Art des RPC-Serververzeichnisses erlaubt es aber nicht, daß auf einem Rechner gleichzeitig mehrere RPC-Server mit gleicher RPC-Programmnummer und gleicher Versionsnummer registriert sind. Im Fall der KS-Server ist dieses jedoch notwendig, da diese alle ein gemeinsames Protokoll (= RPC-Programmnummer) und die gleiche Versionsnummer benutzen.

ACPLT/KS identifiziert statt dessen alle Server auf einem gemeinsamen Rechner über einen Klartextnamen sowie eine Protokoll-Versionsnummer. Die Verwaltung des dazugehörigen

Verzeichnisses der KS-Server übernimmt dabei der KS-Manager (siehe Abbildung 2.1). Bei ihm müssen sich alle KS-Server direkt nach ihrem Programmstart registrieren und später in regelmäßigen Abständen erneut zurückmelden.

2.1 An- und Abmelden eines KS-Servers beim KS-Manager

Die Kommunikation zwischen ACPLT/KS-Servern und ihrem Manager verläuft grundsätzlich nur innerhalb der Maschine, auf der diese Prozesse gestartet wurden. Um weniger Ressourcen für die Server-Manager-Verbindungen zu belegen, wird speziell für diesen Fall UDP/IP eingesetzt. Dieses ist möglich, da nur kurze Anforderungen und Antworten transportiert werden müssen, die in jedem Fall deutlich unterhalb von 8 kByte Größe bleiben (siehe dazu auch [1]).

Die Anmeldephase des Servers beim Manager erfolgt in den nachfolgend beschriebenen Schritten (Abbildung 2.2):

- Der KS-Manager benutzt eine dynamisch vergebene Portnummer. Daher muß der KS-Server im Schritt ① zuerst den (lokalen) Portmapper konsultieren und von diesem die UDP-Portnummer des Managers erfragen.
- Im nächsten Schritt ② ruft der Server (via UDP/IP) den Service `KS_REGISTER` auf, um dem Manager seinen Servernamen, die Serverversion, sowie die benutzte Protokollversionsnummer bekanntzugeben. Daneben gibt der Server bei der Registrierung auch seine Portnummer (TCP-Port) und eine „Rückmeldefrist“ an.
- Vor Ablauf der Rückmeldefrist registriert sich der Server im Schritt ③ erneut über den Service `KS_REGISTER` beim Manager. Auf diese Weise kann der Manager protokollieren, ob die bei ihm registrierten Server zur Zeit noch „funktionstüchtig“ sind. Der Server wiederholt den Schritt ③ in Abständen, solange er läuft. Durch den Vorschlag der Rückmeldefrist durch den Server wird dem möglicherweise unterschiedlichen Antwortverhalten unterschiedlicher Server Rechnung getragen.

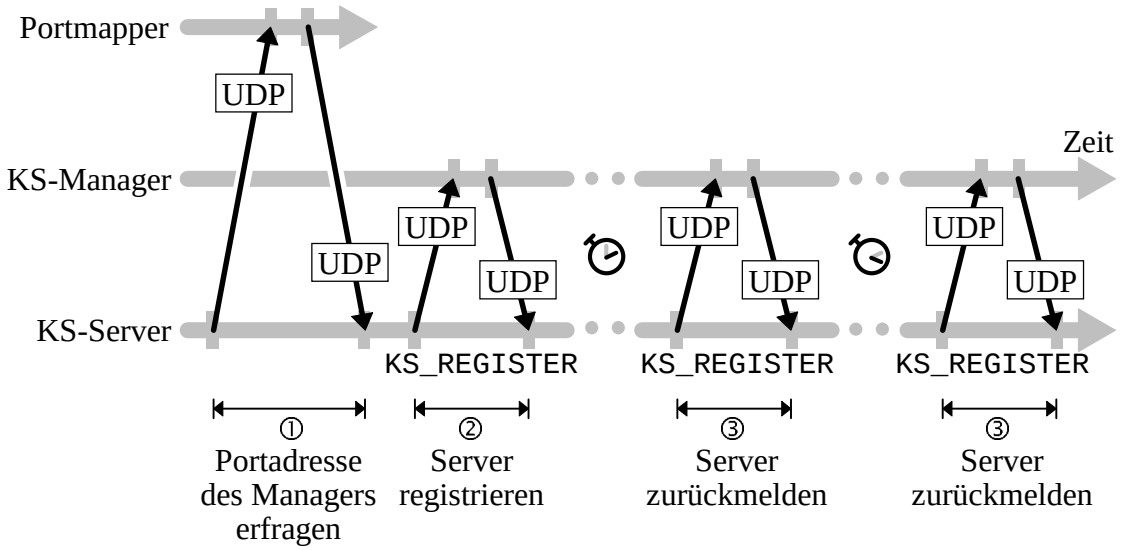


Abbildung 2.2: *Beim Starten eines KS-Servers muß dieser sich zuerst beim KS-Manager anmelden und danach in regelmäßigen Zeitabständen zurückmelden.*

Bevor ein KS-Server endet, sollte er sich noch ordnungsgemäß beim KS-Manager abmelden (Abbildung 2.3, Schritt ④). Hierzu dient der Service `KS_UNREGISTER`. Versäumt ein Server das Abmelden, so wird er zuerst nach Ablauf der Rückmeldefrist vom Manager als inaktiv vermerkt und später dann nach einiger Zeit endgültig aus der Liste der Server entfernt.

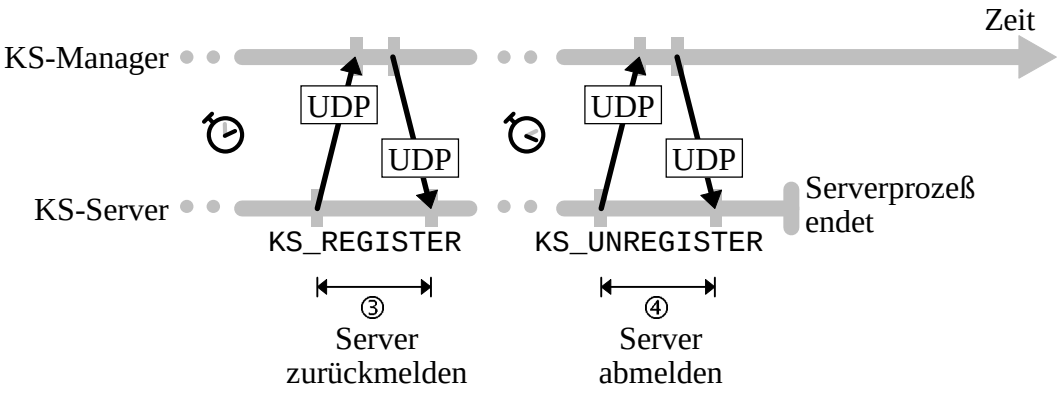


Abbildung 2.3: *Der KS-Server meldet sich bei seinem KS-Manager ab.*

Die beiden Services `KS_REGISTER` sowie `KS_UNREGISTER` werden mit den nachfolgend beschriebenen Parametern aufgerufen und liefern die aufgeführten Antworten zurück.

Durch eine Struktur vom Typ `KS_SERVERDESC` wird ein einzelner Server eindeutig beschrieben. Der Name des Servers (in `name`) darf dabei nur aus den ASCII-Zeichen „A“ bis „Z“, „a“ bis „z“, den Ziffern „0“ bis „9“ sowie dem Unterstrich „_“ bestehen. Andere Zeichen sind nicht zulässig und führen zu der Fehlermeldung `KS_ERR_BADNAME`. Es ist ebenso ein Fehler, die maximal zulässige Länge eines Servernamens von 255 Zeichen zu überschreiten. Groß- und Kleinschreibung im Servernamen wird unterschieden.

```
const KS_NAME_MAXLEN = 255;

struct KS_SERVERDESC {
    string   name<KS_NAME_MAXLEN>;
    u_short  protocol_version;
};
```

Als `protocol_version` gibt ein KS-Server die höchste Versionsnummer des KS-Kernprotokolls an, die er unterstützt. Zur Zeit ist dieses der Wert `0x0001`. Die Struktur `KS_SERVERDESC` wird auch von Klienten benutzt, um den Manager nach einem geeigneten Server zu fragen (Dienst `KS_GETSERVER`). In diesem Fall gibt `protocol_version` diejenige Versionsnummer des KS-Kernprotokolls an, die der Klient vom Server mindestens erwartet (Abbildung 2.4).

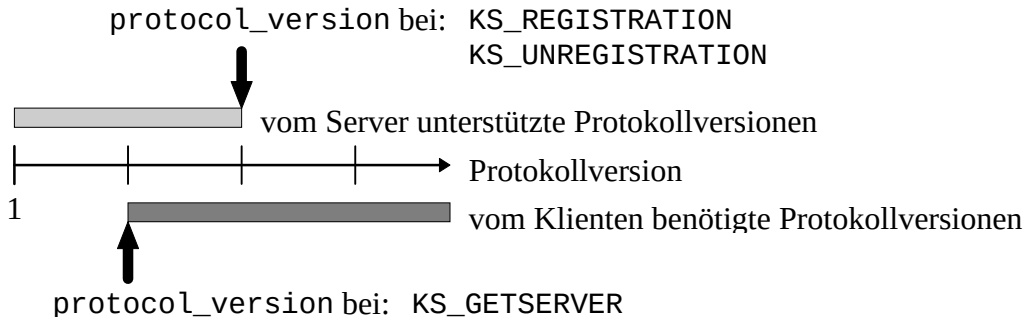


Abbildung 2.4: *Bedeutung der* `protocol_version`.

Hier nun Serviceparameter und -antworten der Dienste `KS_REGISTER` sowie `KS_UNREGISTER` im Detail:

```
struct KS_REGISTRATION_REQ {
    KS_AUTH_REQ    auth;
    KS_SERVERDESC  server;
    u_short        port;
    u_long         time_to_live;
};
```

Die Variable `auth` (vom Typ `KS_AUTH_REQ`) dient der Authentifizierung/Verifizierung. Nähere Einzelheiten hierzu sind in [2] zu finden.

In der Variable `port` gibt der Server bei der Registrierung (beziehungsweise Rückmeldung) die (TCP-) Portnummer an, auf der er Verbindungen von Klienten entgegennimmt. Daneben teilt er dem Manager im Parameter `time_to_live` die gewünschte Frist in Sekunden bis zur nächsten Rückmeldung mit.

```
struct KS_REGISTRATION_REPLY {
    KS_AUTH_REPLY  auth;
    KS_RESULT      result;
};

struct KS_UNREGISTRATION_REQ {
    KS_AUTH_REQ    auth;
    KS_SERVERDESC  server;
};

struct KS_UNREGISTRATION_REPLY {
    KS_AUTH_REPLY  auth;
    KS_RESULT      result;
};
```

Die Services haben die folgenden RPC-Prozedurnummern:

```
KS_REGISTRATION_REPLY KS_REGISTER(KS_REGISTRATION_REQ)       = 0xFF01;
KS_UNREGISTRATION_REPLY KS_UNREGISTER(KS_UNREGISTRATION_REQ) = 0xFF02;
```

Die Services können die folgenden Fehlermeldungen zurückliefern:

`KS_ERR_OK`
: Der Service `KS_REGISTER` oder `KS_UNREGISTER` wurde erfolgreich ausgeführt.

`KS_ERR_GENERIC`
: Es trat bei der Ausführung des Service ein nicht näher definierter Fehler auf.

`KS_ERR_NOREMOTE`
: Der Service darf nur von einem ACPLT/KS-Server angefordert werden, der auf der gleichen Maschine wie der ACPLT/KS-Manager läuft.

`KS_ERR_BADPARAM`
: Ein oder mehrere Parameter sind ungültig. Diese Fehlermeldung tritt auf, wenn `0` als `protocol_version` spezifiziert wurde.

`KS_ERR_BADNAME`
: Der in `name` spezifizierte Servername ist ungültig. Er ist entweder leer oder enthält nicht erlaubte Zeichen.

`KS_ERR_SERVERUNKNOWN`
: Der beim Service `KS_UNREGISTER` in `server` angegebene Server ist nicht registriert.

`KS_ERR_UNKNOWNAUTH`
: Der Klient benutzt in seiner Serviceanforderung ein A/V-Schema, das vom Server nicht unterstützt wird.

KS_ERR_BADAUTH
: Der Klient konnte sich nicht authentifizieren.

2.2 Anmelden eines KS-Klienten bei einem KS-Server

Die „Kontaktaufnahme“ zwischen einem KS-Klienten und KS-Server verläuft zweistufig (Abbildung 2.5) – wenn man die Anfrage beim Portmapper berücksichtigt.

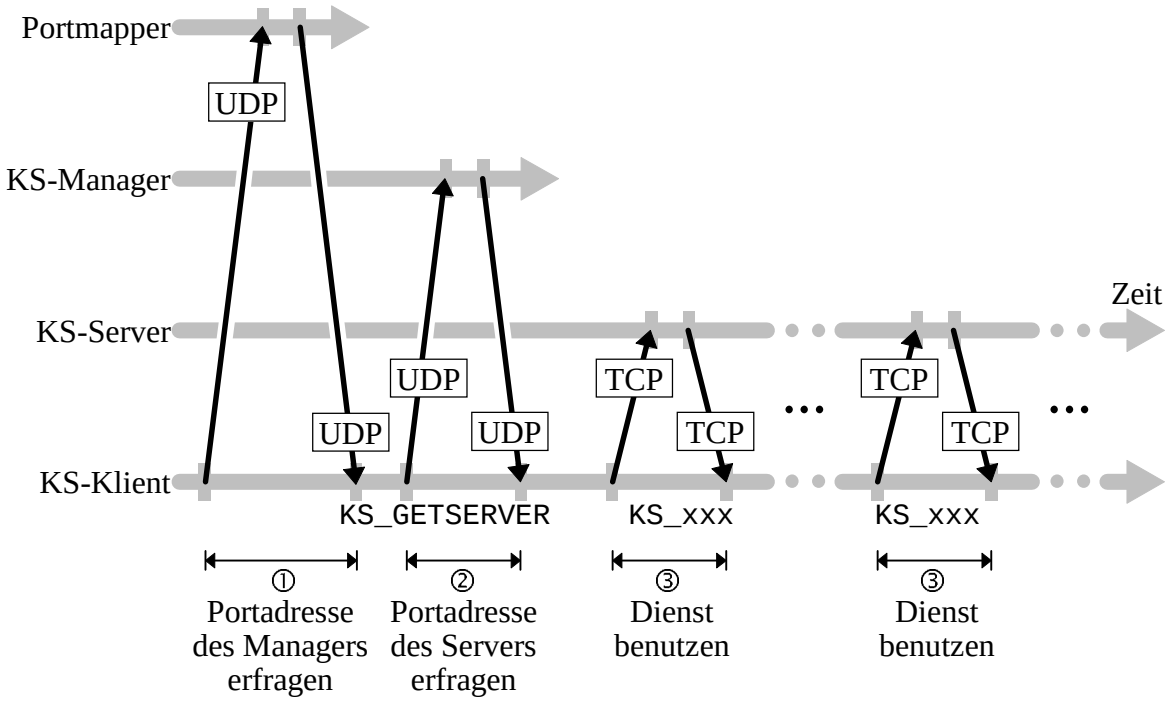


Abbildung 2.5: *Ein KS-Klient erfragt zuerst beim KS-Manager die Portnummer des gewünschten KS-Servers, ehe eine Direktverbindung zwischen Klient und Server zustande kommen kann.*

Die einzelnen Schritte der Anmeldephase sind dabei:

- Da der KS-Manager eine dynamisch vergebene Portnummer benutzt, muß der KS-Klient im Schritt ① zuerst den Portmapper konsultieren und die UDP-Portnummer des Managers erfragen.
- Im nächsten Schritt ② benutzt der Klient den Service `KS_GETSERVER`, um die Portnummer des gewünschten KS-Servers zu ermitteln.
- Mit Hilfe der Portnummer kann der Klient nun eine TCP/IP-Verbindung zum KS-Server aufbauen und Services benutzen (Schritt ③). Sobald der Klient keine weiteren Services beim Servers mehr in Anspruch nehmen möchte, schließt er die TCP/IP-Verbindung. Das ACPLT/KS-Protokoll kennt keine gesonderte Abbauphase der Verbindung – es sind keine besonderen Services aufzurufen, ehe eine Klient-Server-Verbindung geschlossen wird.

Der für den Verbindungsaufbau erforderliche Service `KS_GETSERVER` wird mit den nachfolgend beschriebenen Parametern aufgerufen und liefert die untenstehende Anwort zurück:

```
struct KS_GETSERVER_REQ {
    KS_AUTH_REQ    auth;
    KS_SERVERDESC  server;
};
```

Die Struktur KS_SERVERDESC wurde bereits im vorangehenden Abschnitt „An- und Abmelden eines KS-Servers beim KS-Manager“ erläutert. Für die Variable protocol_version ist diejenige Protokollversion anzugeben, die der Klient mindestens benötigt, um eine Verbindung zum Server aufzubauen.

```
struct KS_SERVER_STATE {
    KS_SERVERDESC  registration;
    u_short        port;
    KS_TIME        expires_at;
    KS_BOOL        living;
};
```

Über das Feld registration der Struktur KS_SERVER_STATE erfährt der Klient die Portnummer, auf der der Server TCP/IP-Verbindungen entgegennimmt. Daneben liefert der Service KS_GETSERVER auch noch Informationen über den Zeitpunkt, an dem sich der Server spätestens wieder beim Manager zurückmelden sollte und ob der gewünschte Server bereits diesen Zeitpunkt versäumt hat.

```
union KS_GETSERVER_RET switch (KS_RESULT result) {
    case KS_ERR_OK:
        KS_SERVER_STATE  server_state;
    default:
        void;
};

struct KS_GETSERVER_REPLY {
    KS_AUTH_RES      auth;
    KS_GETSERVER_RET ret;
};
```

Der Service hat die folgende RPC-Prozedurnummer:

```
KS_GETSERVER_RES KS_GETSERVER(KS_GETSERVER_REQ) = 0xFF03;
```

Der Service kann die folgenden Fehlermeldungen zurückliefern:

KS_ERR_OK
: Der Service KS_GETSERVER wurde erfolgreich ausgeführt.

KS_ERR_GENERIC
: Es trat bei der Ausführung des Service ein nicht näher definierter Fehler auf.

KS_ERR_BADPARAM
: Ein oder mehrere Parameter sind ungültig. Dieser Fehler tritt auf, wenn als gewünschte ACPLT/KS-Protokollversion in protocol_version der Wert 0 angegeben wurde.

KS_ERR_BADNAME
: Der in name spezifizierte Servername ist ungültig. Er ist entweder leer oder enthält nicht erlaubte Zeichen.

KS_ERR_SERVERUNKNOWN
: Der beim Service KS_GETSERVER in server angegebene Server ist nicht registriert.

KS_ERR_UNKNOWNAUTH
: Der Klient benutzt in seiner Serviceanforderung ein A/V-Schema, das vom Server nicht unterstützt wird.

KS_ERR_BADAUTH

Der Klient konnte sich nicht authentifizieren.

3 AuA – Abkürzungen und Akronyme

ACPLT/KS	Kommunikationssystem des Lehrstuhls für Prozeßleittechnik der RWTH Aachen
IP	Internet Protocol
ISO	International Standards Organization
ONC	Open Network Computing
OSI	Open Systems Interconnect
RPC	Remote Procedure Call/Calling
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
XDR	External Data Representation

4 Literaturverzeichnis

[1] ACPLT/KS Group:
Technologiepapier #2: Der Nachrichtentransport.
Lehrstuhl für Prozeßleittechnik, RWTH Aachen, 1996

[2] ACPLT/KS Group:
Technologiepapier #7: A/V-Module.
Lehrstuhl für Prozeßleittechnik, RWTH Aachen, 1996