OPROGRAMOWANIE KOMUNIKACYJNE

Wojciech Gumiński

http://guminski.net

wojciech.guminski@eti.pg.gda.pl

Oprogramowanie komunikacyjne

Zakres zagadnień poruszanych na wykładzie

- Definicja i cechy oprogramowania komunikacyjnego
- Proces projektowania oprogramowania komunikacyjnego
- Środowisko oprogramowania komunikacyjnego
- Przykłady specyfikacji nieformalnej
- SDL – specification and description language
- Przykłady specyfikacji formalnej

DEFINICJA OPROGRAMOWANIA KOMUNIKACYJNEGO

OPROGRAMOWANIE KOMUNIKACYJNE

Pojęcie oprogramowania komunikacyjnego nie jest jednoznacznie zdefiniowane.

Na potrzeby tego wykładu będzie przyjęta następująca definicja:

Oprogramowanie komunikacyjne to oprogramowanie, które:

- **realizuje komunikację między komputerami,**
- **wspomaga komunikację między ludźmi,**
- **zarządza obiema formami komunikacji.**

Cechy Oprogramowania komunikacyjnego

Cechy wspólne z innymi typami oprogramowania:

- pożądane **niskie koszty wytwarzania.**

Cechy wspólne z innymi typami oprogramowania, ale o szczególnym znaczeniu dla oprogramowania komunikacyjnego:

- **otwartość na nowe usługi i rozwiązania** (ewolucyjność)
  - konstrukcja oprogramowania powinna umożliwiać nadążanie za zmianami w technologii oraz w rozbudowie funkcjonalności oprogramowania.
- **duża złożoność** i **wielkość oprogramowania**
  - praca w czasie rzeczywistym
  - praca współbieżna
  - praca rozproszona

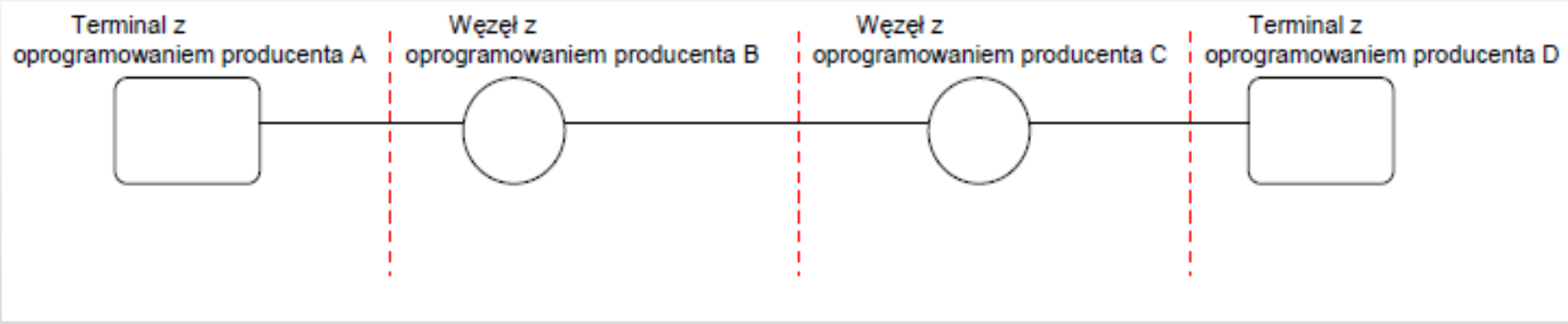
CECHY OPROGRAMOWANIA KOMUNIKACYJNEGO

- **długi czas życia** oprogramowania narzucający **wysokie wymagania jakościowe** na:
  - oprogramowanie oraz
  - jego dokumentację
  - oraz powodujący **zmienność zespołu programistów** nawet u jednego producenta oprogramowania. Czas życia oprogramowania komunikacyjnego obliczany jest na okres rzędu 20 lat, bez względu na to czy jest to oprogramowanie dla sieci telekomunikacyjnych, czy też dla sieci komputerowych. Wynika to z natury sieci komunikacyjnych, w których elementem przesądzającym o czasie korzystania z określonych usług jest nie tyle jej nowoczesność co powszechność.

Cechy Oprogramowania komunikacyjnego

Cecha, która nie jest istotne przy projektowaniu innych typów oprogramowania, ale jest zasadnicza przy projektowaniu oprogramowania komunikacyjnego

- **różni wytwórcy** oprogramowania
  - oprogramowanie musi ściśle i bezbłędnie ze sobą współpracować (dobrym przykładem są protokoły komunikacyjne).



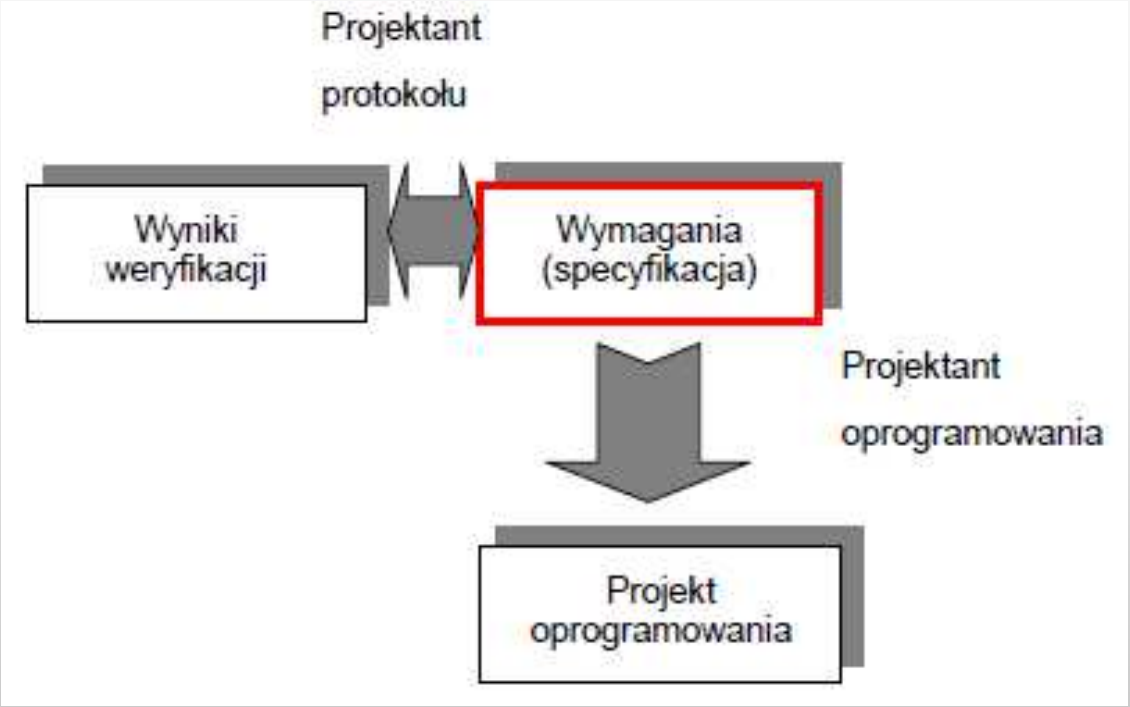
PROJEKTOWANIE OPROGRAMOWANIA KOMUNIKACYJNEGO

PROJEKTOWANIE OPROGRAMOWANIA KOMUNIKACYJNEGO

Pojęcie specyfikacji i języka specyfikacji:

- Rysunek techniczny konstrukcji mechanicznej
- Schemat ideowy układu elektronicznego
- Opis algorytmu numerycznego

Specyfikacja to sformułowanie wymagań co do sposobu funkcjonowania urządzenia, bądź programowania.



Dlaczego używamy języków specyfikacji?

- do komunikacji między:
  - producentami
  - projektantami
  - programistami
- jako narzędzia automatycznego wspomagania procesu wytwarzania oprogramowania
  - realizacja
  - weryfikacja
  - dokumentacja

PROJEKTOWANIE OPROGRAMOWANIA KOMUNIKACYJNEGO

Rozpatrując proces projektowania oprogramowania komunikacyjnego należy wziąć pod uwagę dwa punkty widzenia:

- **punkt widzenia użytkownika oraz**
- **punkt widzenia projektanta oprogramowania.**

To spojrzenie wskazuje, że **klient** ma w kontekście projektowania oprogramowania komunikacyjnego mniejsze znaczenie. Punkt widzenia użytkownika związany jest z długim okresem eksploatacji systemu, w którym to okresie zmieniają się wymagania funkcjonalne i ruchowe na system.

Punkt widzenia projektanta jest zdeterminowany wielkością i różnorodnością zadań, które ma to oprogramowanie realizować oraz niepełną znajomością przyszłych wymagań.

PROJEKTOWANIE OPROGRAMOWANIA KOMUNIKACYJNEGO

- Użytkownik
  - Jakość funkcjonowania i utrzymania:
  - · łatwość użytkowania,
  - · modyfikowalność funkcji,
  - · modyfikowalność obciążenia,
  - · niezawodność funkcjonowania.
  - Czas zrealizowania

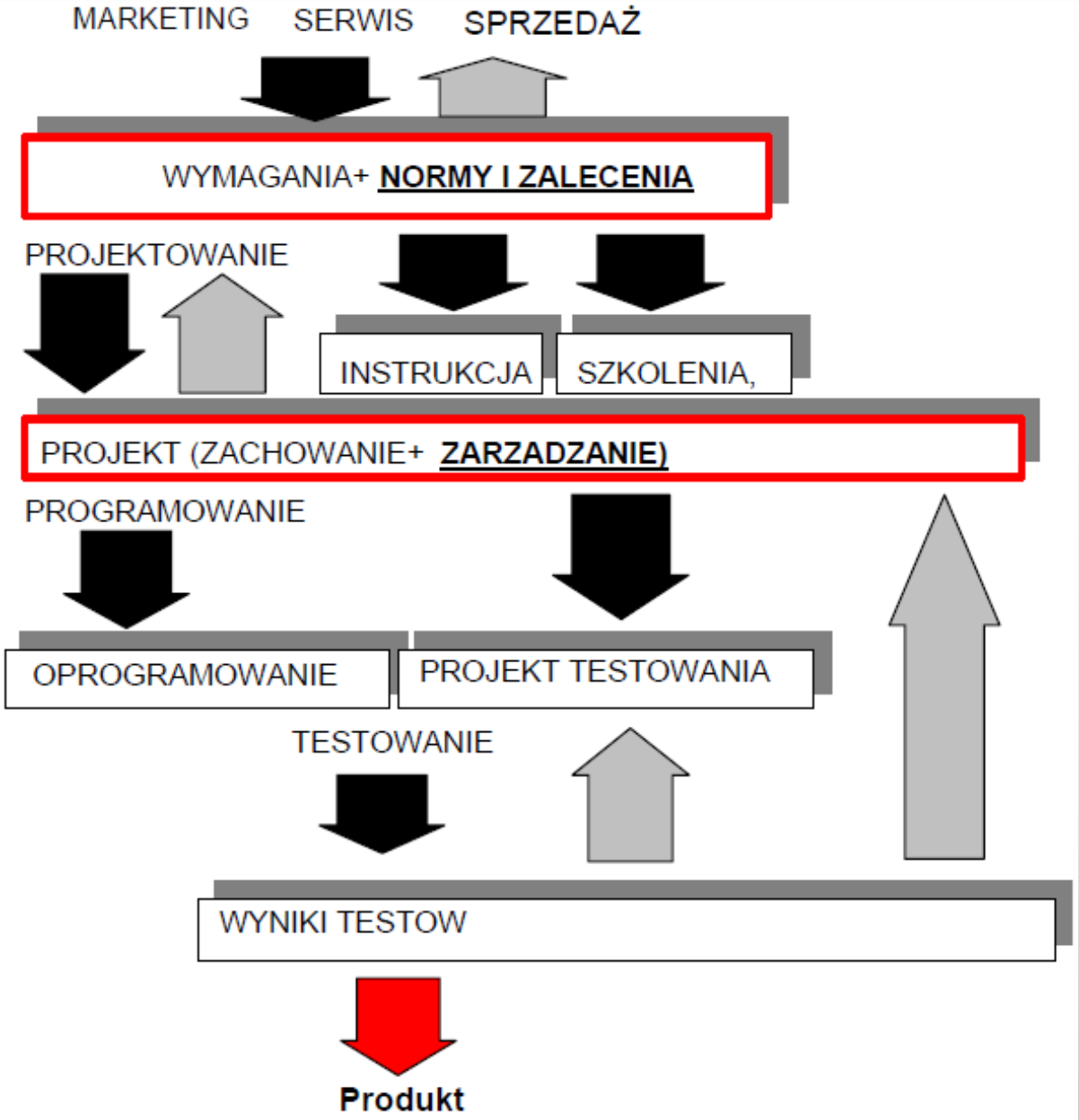
- Projektant
  - Jakość realizowania:
  - · łatwość oprogramowywania,
  - · łatwość uruchamiania i testowania,
  - · modyfikowalność oprogramowania.
  - Czas zrealizowania

PROJEKTOWANIE OPROGRAMOWANIA KOMUNIKACYJNEGO

Jak widać oczekiwania obu stron nie zawsze są sprzeczne. Wymaganie modyfikowalności oferowanych funkcji stoi co prawda w sprzeczności z łatwością realizacji oprogramowania, lecz nie stoi w sprzeczności z wymaganiem modyfikowalności oprogramowania. Podstawą spełnienia oczekiwań obu stron jest posiadanie:

- jednolitego sposobu:
  - formułowania wymagań
  - projektowania oprogramowania,
  - implementacji oprogramowania,
  - testowania oprogramowania i jego weryfikacji.
- modelu działania oraz metod analizy wykorzystanych algorytmów sterowania i komunikacji

PROJEKTOWANIE OPROGRAMOWANIA KOMUNIKACYJNEGO



PROJEKTOWANIE OPROGRAMOWANIA KOMUNIKACYJNEGO

Wymagania techniczno-ekonomiczne są opracowywane by:

- być merytoryczną podstawą do podejmowania decyzji: poprawiających rentowność produkcji, zmniejszenia liczby działań zbędnych, ryzykownych lub mało rentownych lub zapewniających korzystniejsze rozłożenie w czasie podejmowanych działań,
- stworzyć pośrednika pomiędzy projektowaniem oprogramowania i sprzętu, sprzedażą i marketingiem oraz serwisem, który to pośrednik byłby zrozumiały dla wszystkich stron,
- sprecyzować, uporządkować i zorganizować zadania dla zespołów projektowych, dokumentalistów (wykonanie instrukcji), serwisu i sprzedaży po podjęciu decyzji o rozpoczęciu procesu produkcyjnego,
- łatwiej uchwycić i wyegzekwować odpowiedzialność za efekty oraz ocenić odpowiedzialnych za nie.

PROJEKTOWANIE OPROGRAMOWANIA KOMUNIKACYJNEGO

Wymagania techniczno-ekonomicznych opracowuje się na podstawie:

1. funkcjonujących standardów opisujących protokoły i usługi komunikacyjne.
   - dokumenty normalizacyjnych (ISO, IEEE, ITU itp),
   - książki i literatura oraz informacji o książkach i literaturze poświęconej protokołom i usługom,
   - dokumentacje innych producentów,
   - udziału (biernego i aktywnego) w pracach organizacji normalizacyjnych np. ATM Forum,
2. życzeń klientów
   - analiza rynku usług komunikacyjnych,
   - kreowanie potrzeb rynku,
   - bezpośrednie kontakty z klientami i dystrybutorami
   - zbieranie uwag od własnego serwisu i laboratorium wykonującego testy

3. własnych propozycji i rozwiązań problemów nowych i nierozwiązanych lub nieopisanych
4. analizy produktów oferowanych przez innych producentów,

- czasopisma, pokazy, akcje promocyjne, funkcjonujące instalacje
- porównanie specyfikacji technicznych produktów,
- dostęp do wybranych produktów.

Zawartość wymagań to:

- charakterystyki techniczne i funkcjonalne proponowanych produktów
- oszacowania kosztów ich wdrożenia obejmujące cały cykl produkcyjny
- oszacowania kosztów eksploatacji usług u użytkownika

PROJEKTOWANIE OPROGRAMOWANIA KOMUNIKACYJNEGO

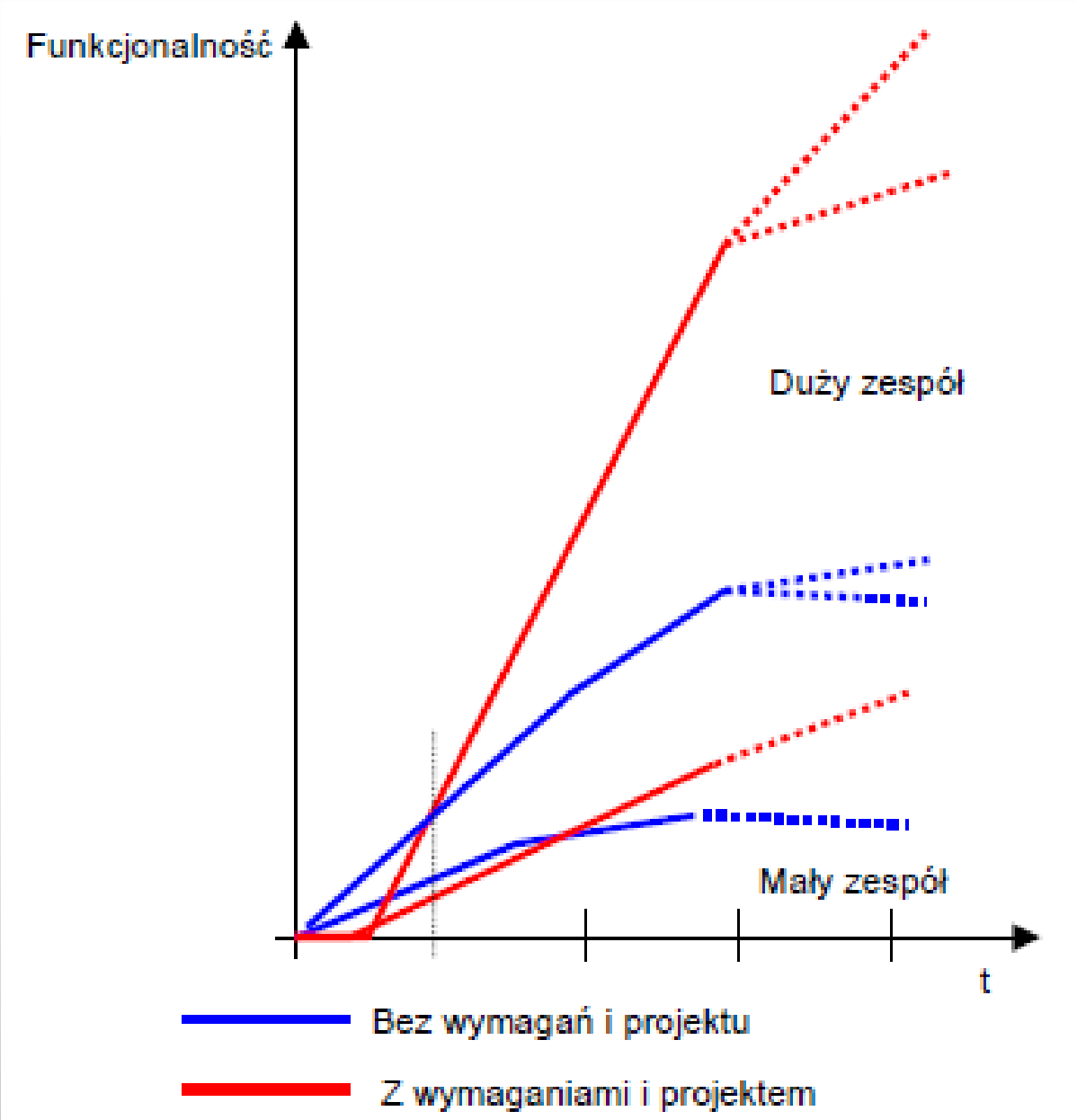
Ze względu na funkcjonowanie norm i zaleceń oraz specyfikę oprogramowania komunikacyjnego bardzo często klient nie dysponuje odpowiednimi kwalifikacjami do wykonania wymagań. W takim przypadku powinien zlecić ich wykonanie, a nie pominąć etap w cyklu wytwarzania oprogramowania (co w praktyce ma miejsce nadzwyczaj często). Pominięcie tego elementu cyklu jest najczęstszym źródłem przedłużenia cyklu i wzrostu kosztów. Stosunkowo często wymagania powstają jako suma (złożenie) funkcjonalności oferowanych przez innych producentów, bez wnikania w sposoby uzyskania tej funkcjonalności oraz możliwości ich realizacji. Prowadzi to do podobnych efektów jak brak wymagań.

PROJEKTOWANIE OPROGRAMOWANIA KOMUNIKACYJNEGO

Projekty są opracowywane by:

- zmniejszyć koszty realizacji zadań projektantów poprzez:
- zmniejszenie liczby błędów i wysokich kosztów ich usuwania w chwili, gdy ujawnią się one w gotowym wyrobie (patrz rozkład Pareto - poprawienie 20% błędów pochłania 80% czasu),
- zapewnienie właściwej struktury zatrudnienia polegającej na wydzieleniu przynajmniej dwóch grup pracowników inżynieryjnych:
  - 1. nielicznej grupy doświadczonych, lecz kosztownych projektantów i
  - 2. liczniejszej grupy tańszych programistów.
- urealnienie ocen kosztów wdrożenia projektu.
- efektywnie i bezpiecznie powiększyć zakresu możliwości funkcjonalnych wyrobów.
- zrównoleglić z procesem programowania działania innych grup np. opracowujących testy, dokumentację, sprzedaż lub szkolenia.

PROJEKTOWANIE OPROGRAMOWANIA KOMUNIKACYJNEGO



PROJEKTOWANIE OPROGRAMOWANIA KOMUNIKACYJNEGO

Podstawowymi wymaganiami stawianymi wobec metod formalnej specyfikacji protokołów są:

- niedwuznaczna, precyzyjna, kompletna i niezależna od sposobu realizacji definicja standardu,
- czytelne źródło odniesienia dla użytkowników, realizatorów i sprawdzających zgodność ze standardem,
- formalnie dobrze zdefiniowana podstawa do weryfikacji, walidacji i sprawdzania zgodności ze standardem.

PROJEKTOWANIE OPROGRAMOWANIA KOMUNIKACYJNEGO

Właściwości, które musi posiadać metoda formalnej specyfikacji:

- **moc wyrażania**; metoda musi gwarantować wyrażenie szerokiego zakresu obiektów i ich właściwości, które są niezbędne do opisu usług i protokołów,
- **definicje formalne;** składnia i semantyka metody powinna mieć kompletną i formalną definicję. Model formalny leżący u podstaw semantyki musi zabezpieczać rozwój teorii analitycznej, która może być użyta do weryfikacji walidacji i testów zgodności:
- **abstrakcyjność**; konstrukcja językowa powinna reprezentować stosowną koncepcję architektoniczną na dostatecznie wysokim poziomie abstrakcji, który będzie wolny od realizacyjnie zorientowanych szczegółów. Pozwoli to uniknąć ograniczeń realizacyjnych.

PROJEKTOWANIE OPROGRAMOWANIA KOMUNIKACYJNEGO

Właściwości, które musi posiadać metoda formalnej specyfikacji:

- **strukturalność;** metoda powinna oferować środki do strukturalizacji wymagań w taki sposób by zapewnić ich zrozumiałość i przejrzystość. Strukturalizacja powinna umożliwić kompresję nieistotnych szczegółów na każdym poziomie projektowania tak by zredukować lokalną złożoność fragmentów opisu oraz liczbę decyzji podejmowanych na każdym poziomie (i w ten sposób liczbę potencjalnych błędów). Dobra struktura opisu będzie gwarancją jego czytelności, elastyczności i prostoty w utrzymaniu. Stanowi też dobrą podstawę do efektywnej analizy opisu.

PROJEKTOWANIE OPROGRAMOWANIA KOMUNIKACYJNEGO

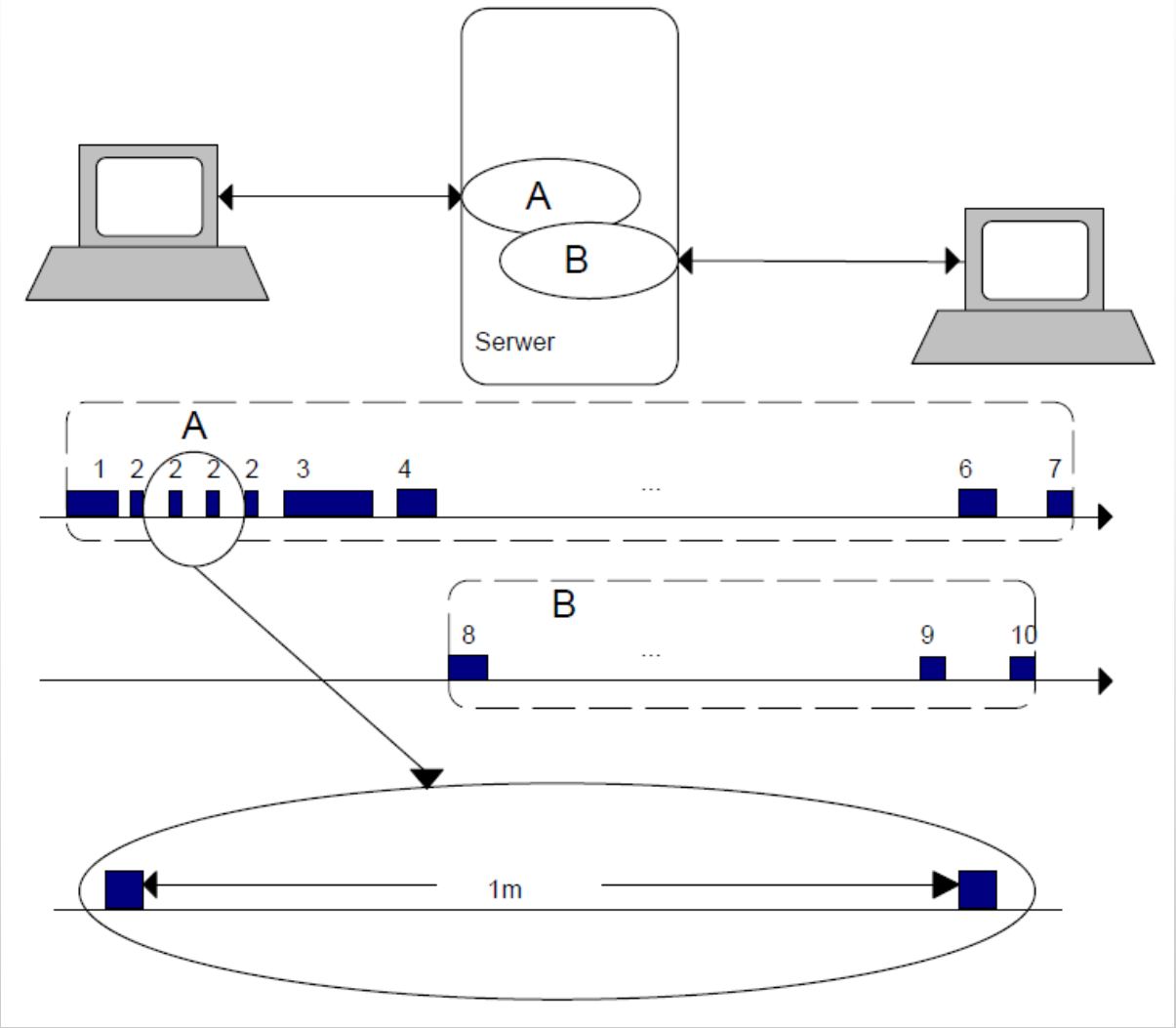
Cechy te są szczególnie istotne, gdy należy projektować rzeczywiste protokoły i usługi, które charakteryzują się dużą złożonością struktury algorytmicznej i komunikacyjnej. Nieposiadanie tych cech przez metodę może powodować powstawanie błędów możliwych do usunięcia już tylko na poziomach charakteryzujących się wysokimi kosztami wprowadzania poprawek i zmian.

Właściwości te muszą posiadać również metody formalnej specyfikacji wymagań dla standardów ISO/OSI (Open System Interconnection - łączenia systemów otwartych).

ŚRODOWISKO OPROGRAMOWANIA KOMUNIKACYJNEGO

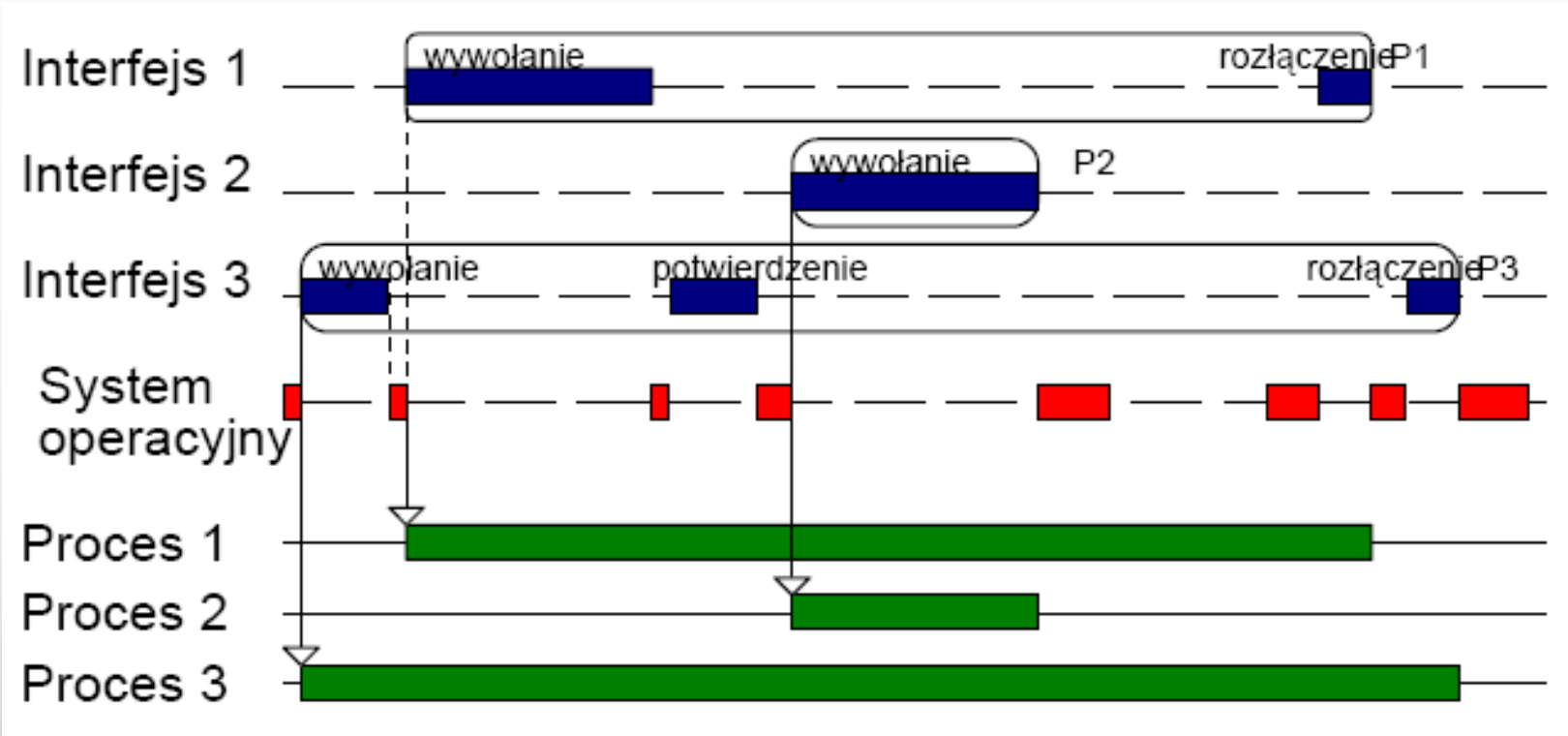
ŚRODOWISKO OPROGRAMOWANIA KOMUNIKACYJNEGO

Środowisko oprogramowania komunikacyjnego to różne formy środowiska współbieżnego.

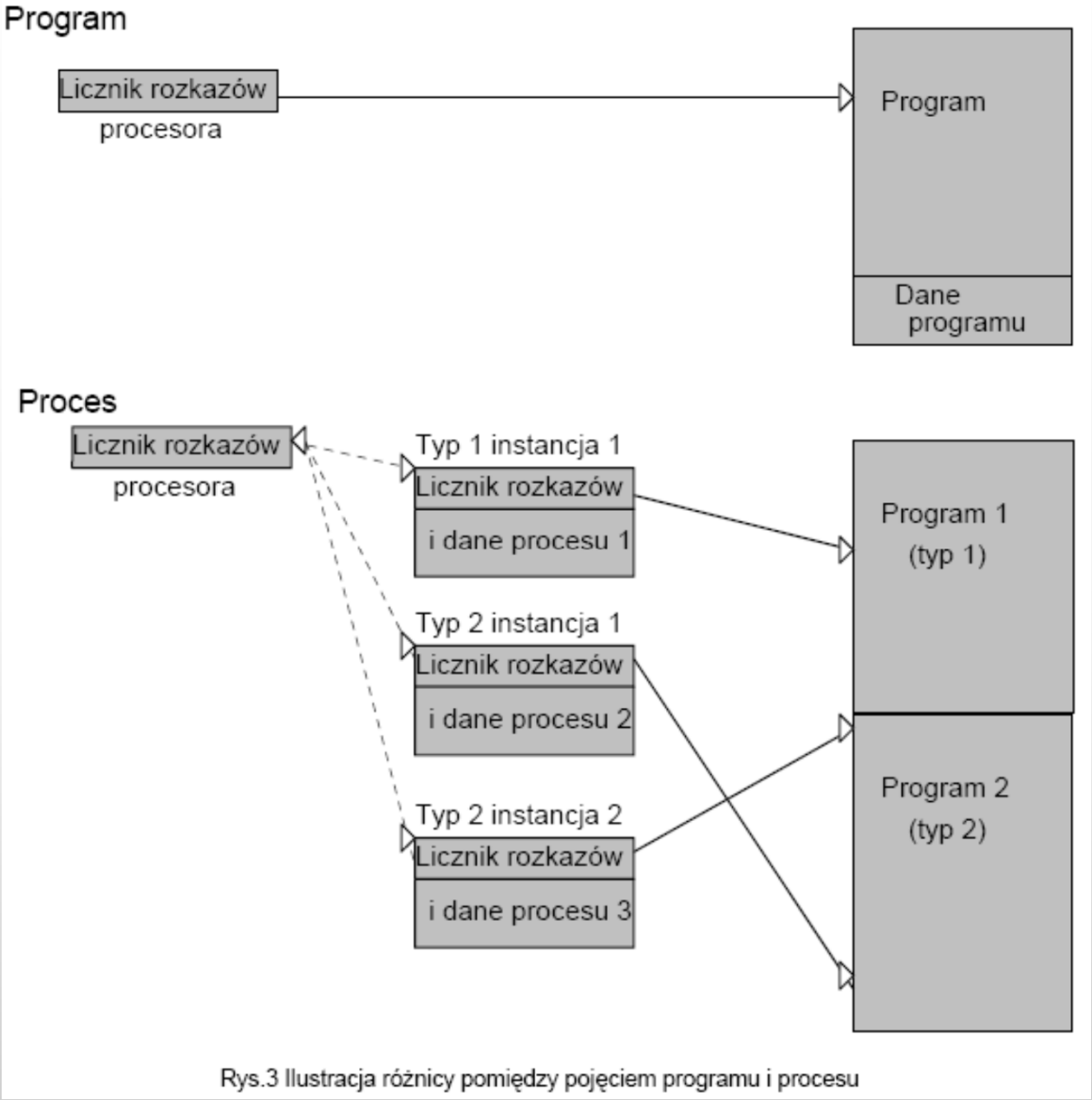


Środowisko oprogramowania komunikacyjnego

- **Proces** (instancja procesu) jest ciągiem wykonań instrukcji jego programu,
- wyznaczonych kolejnymi wartościami jego licznika rozkazów, wraz z jego
- danymi, wskaźnikiem stosu i stosem.



Środowisko oprogramowania komunikacyjnego



Rys.3 Ilustracja różnicy pomiędzy pojęciem programu i procesu

Środowisko oprogramowania komunikacyjnego

Stany procesu

Procesy mogą znajdować się w następujących stanach:

- **Stan wykonywania**

  W stanie wykonywania w danej chwili może znajdować się tylko jeden proces. Jest to stan, w którym proces posiada procesor.

- **Stan gotowości**

  Procesy znajdujące się w stanie gotowym przeznaczone są do wykonania, gdy tylko ich priorytet będzie dostatecznie wysoki. Są to procesy oczekujące tylko na zwolnienie procesora.

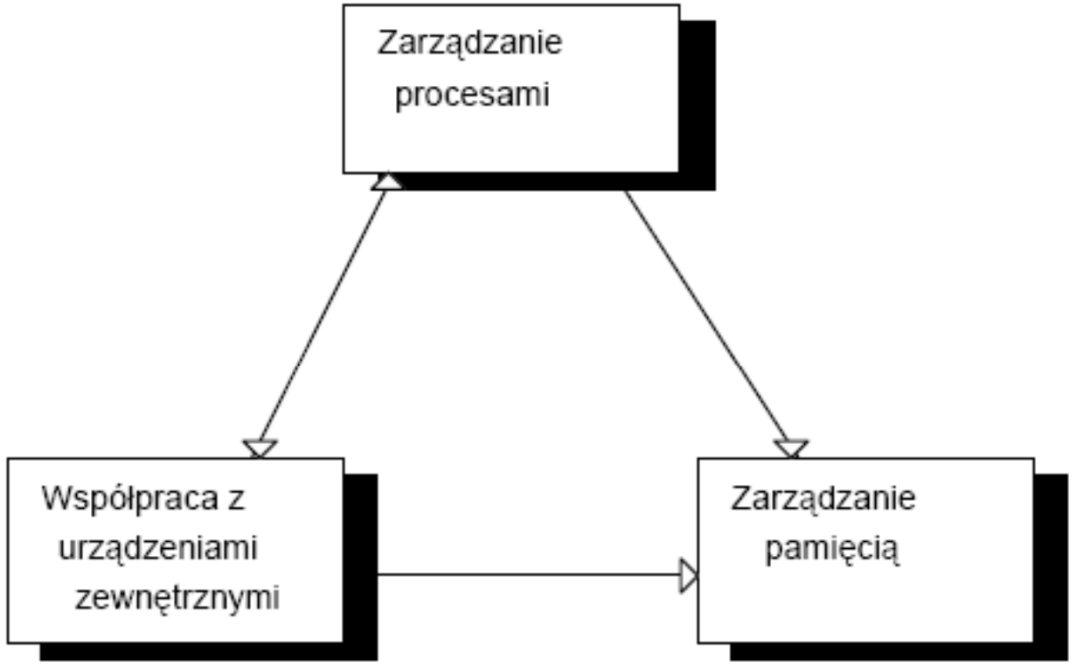
- **Stan zablokowania (zawieszenia)**

  Procesy zawieszone oczekują na spełnienie warunków np. upływu czasu, dostępu do wspólnych zasobów znajdują się one w stanie oczekiwania.

ŚRODOWISKO OPROGRAMOWANIA KOMUNIKACYJNEGO

Rys. 4 Graf przejść pomiędzy stanami procesu

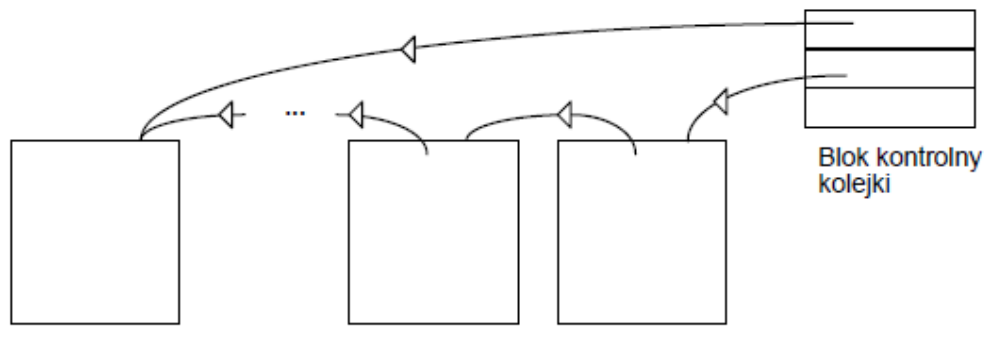
Środowisko oprogramowania komunikacyjnego

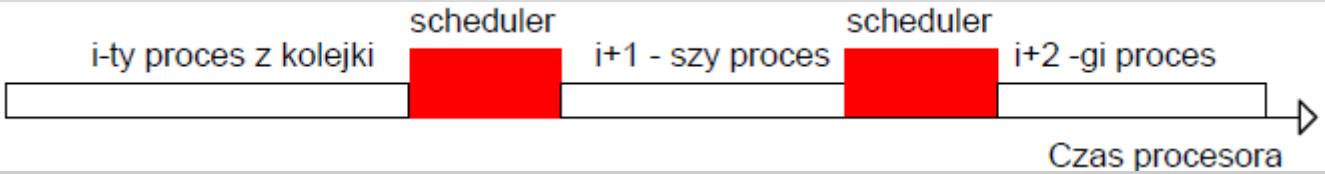


Rys. 5 Elementy systemu operacyjnego

Przydział czasu procesora

Podstawa przydziału procesora jest kolejka procesów gotowych (deskryptorów tych procesów) oczekujących na procesor. Kolejka jest obsługiwana przez proces nazywany schedulerem (planistą). Proces ten pobiera z kolejki deskryptory procesów według zasad charakteryzujących scheduler i przydziela pobranemu procesowi procesor.
W podobnych kolejkach oczekują na spełnienie warunków dalszej realizacji deskryptory procesów zablokowanych.





ŚRODOWISKO OPROGRAMOWANIA KOMUNIKACYJNEGO

Wywłaszczenie procesu - odebranie procesowi procesora w wyniku zajścia zdarzenia zewnętrznego w stosunku do procesu.

Algorytmy szeregowania procesów

1. kolejkowy bez wywłaszczania
2. okrężny (round-robin) (z wywłaszczaniem)
3. priorytetowy - statyczny i dynamiczny (bez wywłaszczania i z wywłaszczaniem)

Wady i zalety tych algorytmów:

ad. 1.

- - nieznany czas oczekiwania na procesor
- + prostota

ad. 2.

- - wywłaszczanie
- - niesprawiedliwość

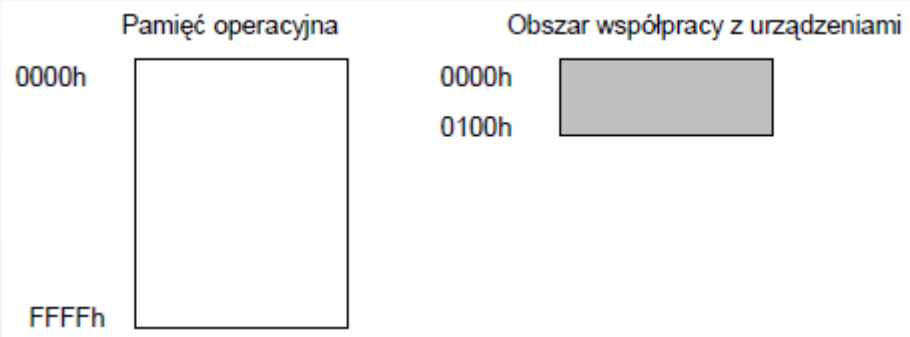
ad.3.

- - złożoność
- - wywłaszczanie
- + sprawiedliwość

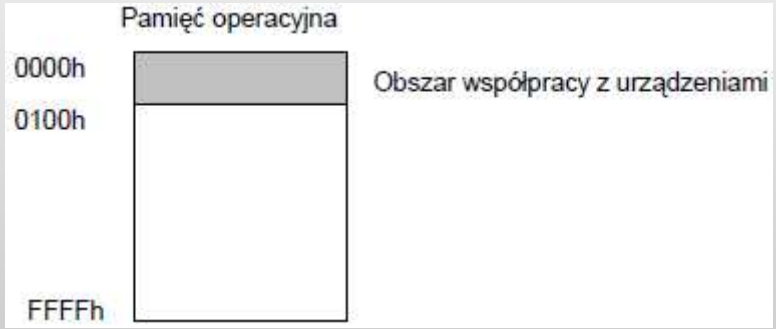
Zasady współpracy z urządzeniami zewnętrznymi

Dostęp do urządzeń komunikacyjnych

- Korzystanie ze specjalnych rozkazów procesora do dostępu do urządzeń zewnętrznych



- I/O Mapping – zmniejszenie przestrzeni adresowej pamięci procesora



ŚRODOWISKO OPROGRAMOWANIA KOMUNIKACYJNEGO

Polling – cykliczne przeglądanie rejestrów urządzeń zewnętrznych

Technika przydatna tylko w przypadku niewielkiej liczby urządzeń o w miarę równomiernym czasie obsługi bądź w systemach synchronicznych o ustalonej kolejności i czasie obsługi.

Interrupt - przerwania

Technika polegająca na generowaniu przez urządzenie zewnętrzne sygnału informującego o zmianie stanu tego urządzenia.
Z przerwaniami można powiązać priorytet przerwania. Dwie najczęściej stosowane techniki to:

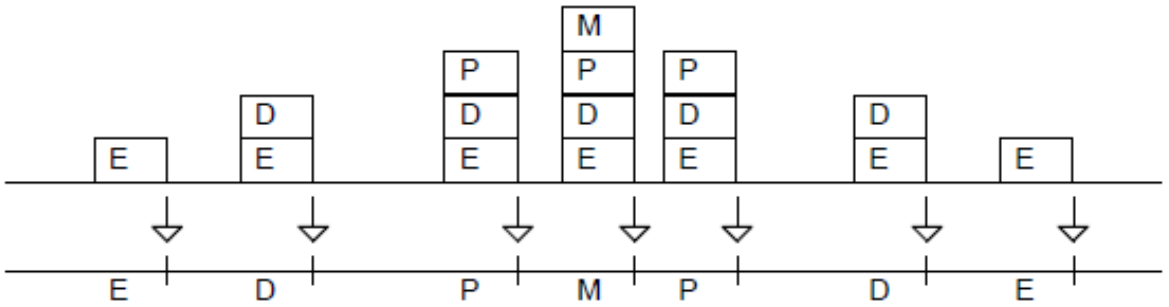
- łączenie generatorów przerwań w łańcuchy
- przypisanie procesorowi dodatkowego atrybutu (cechy) określającego jego priorytet w stosunku do urządzeń zewnętrznych

DMA – bezpośredni dostęp do pamięci

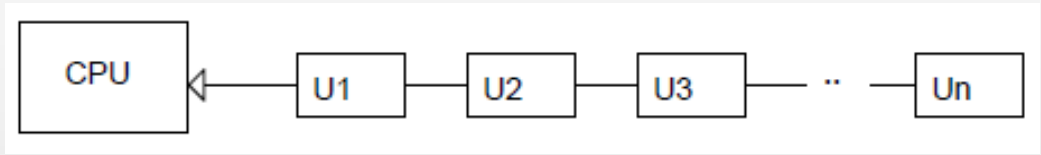
Technika polegająca na bezpośredniej obsłudze pamięci systemowej przez sterownik urządzenia zewnętrznego. DMA najczęściej wykorzystywane jest przez szybkie urządzenia przekazujące duże ilości danych.

ŚRODOWISKO OPROGRAMOWANIA KOMUNIKACYJNEGO

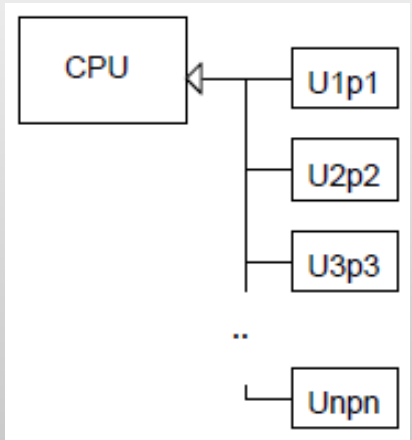
Przykład: Sekwencja przerwań od dysku, drukarki i modemu



Łańcuch przerwań

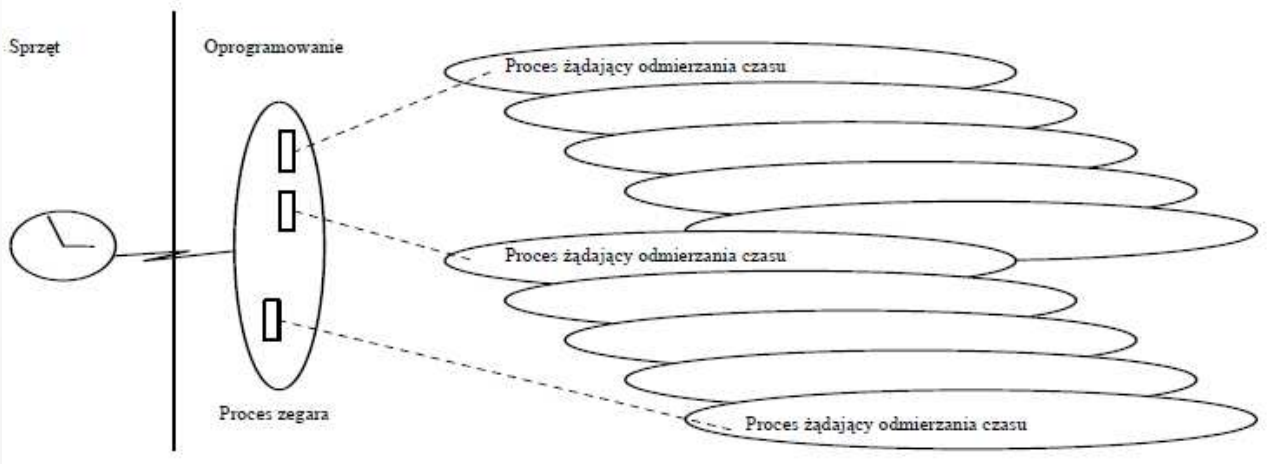


System przerwań z programowymi priorytetami

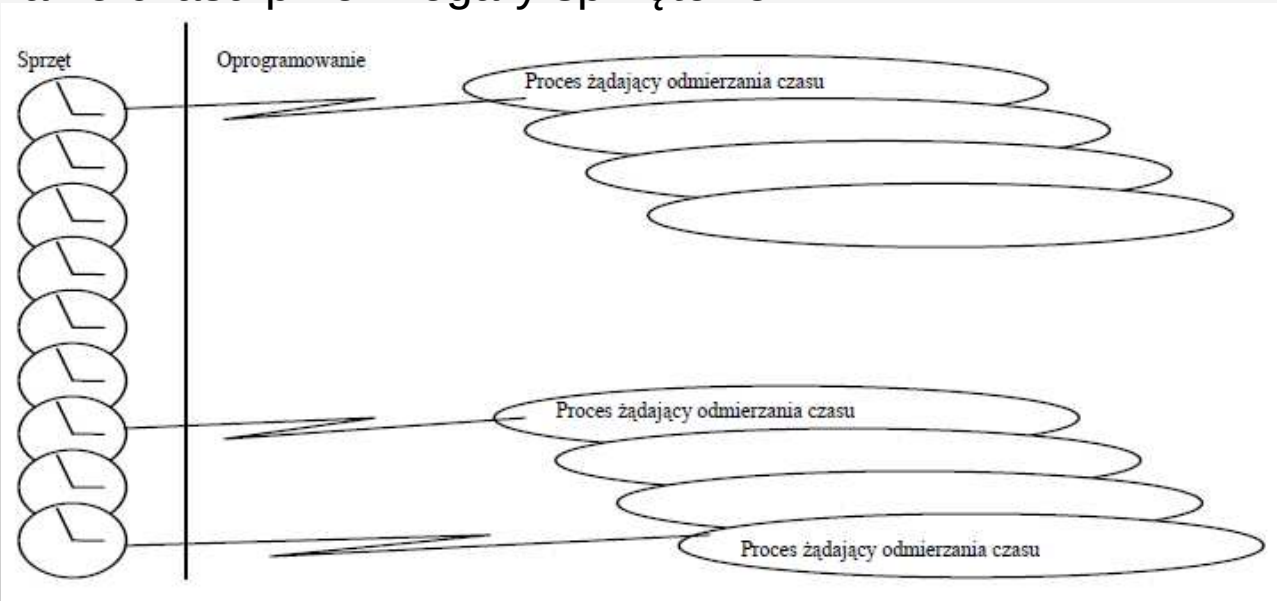


ŚRODOWISKO OPROGRAMOWANIA KOMUNIKACYJNEGO

Odmierzanie czasu przez proces zegara



Odmierzanie czasu przez zegary sprzętowe



PRZYKŁADY SPECYFIKACJI NIEFORMALNEJ

Protokół kolejowy

(Katastrofa kolejowa spowodowana wadliwą konstrukcją protokołu.)

Przypadki wypadków kolejowych są zazwyczaj przebadane i udokumentowane z dokładnością do minuty, tak, że ...

Pojedynczy przykład powinien wystarczyć do zilustrowania jak ważne katastrofy mogły być wynikiem niespodziewanej wcześniej kombinacji zdarzeń.

SPECYFIKACJE NIEFORMALNE

Wypadku, który będzie dalej opisany, można by było uniknąć gdyby użyto odpowiedniego protokołu do zorganizowania komunikacji między dróżnikami.

Wypadek ten zdarzył się w tunelu Clayton, który wydawał się być najlepiej chronionym szlakiem kolejowym w Anglii.

Na każdym końcu tunelu o długości ponad 2 km, przez 24 godziny na dobę, pełnili służbę dróżnicy. Dodatkowo w roku 1841 tunel wyposażono w nowy system sygnalizacji. Zbudowany on był z semaforów na każdym końcu tunelu. System działał w ten sposób, że każdy pociąg mijający semafor zezwalający na wjazd do tunelu zmieniał jego położenie na sygnał zakazujący wjazdu do tunelu. Obowiązkiem dróżnika było ponowne ustawienie semafora na pozycję zezwalającą na wjazd. Dróżnik ten zanim zmienił ustawienie semafora miał obowiązek upewnić się, że pociąg, który wjeżdżał do tunelu opuścił go.

SPECYFIKACJE NIEFORMALNE

W celu usprawnienia pracy dróżników tunel wyposażono w telegraf igłowy. System ten dobudowano by możliwość wymianę między dróżnikami kilku wcześniej zdefiniowanych komunikatów. Po udzieleniu zgody na wjazd pociągu do tunelu dróżnik nadawał wiadomość ***pociąg w tunelu*** do dróżnika na drugim końcu tunelu. Gdy pociąg wyjeżdżał z tunelu dróżnik wysyłał wiadomość ***tunel pusty*** . Po otrzymaniu tego sygnału pierwszy dróżnik mógł ustawić semafor tak by umożliwić wjazd następnego pociągu. Aby system uczynić całkowicie pewnym, dodano trzeci komunikat: ***czy pociąg wyjechał z tunelu?*** Obecność dwóch dróżników gwarantowała, że tunel można było bezpiecznie używać nawet wtedy, gdy z jakiś powodów, semafory zadziałałyby niewłaściwie. Jeśli pociąg wjeżdżał do tunelu przy braku prawa wjeżdżania, dróżnik był o tym informowany za pomocą dzwonka. W takim przypadku dróżnik mógł za pomocą białej i czerwonej flagi sterować ruchem. Jednak okazało się,że protokół nie był wystarczająco dobrze opisany.

SPECYFIKACJE NIEFORMALNE

Oto co się zdarzyło w sierpniu 1861 roku.

- Pierwszy pociąg przejechał semafor i nie zmienił położenia semafora. Zgodnie z przewidywaniami dzwonek ostrzegł dróżnika. Zgodnie z procedurą dróżnik najpierw wysłał do dróżnika z drugiej strony tunelu wiadomość ***pociąg w tunelu*** i następnie udał się ustawić czerwoną flagę tak by zatrzymać następny pociąg.
- Niestety drugi pociąg nadjechał szybciej niż się spodziewał dróżnik i wjechał do tunelu zanim dróżnik ustawił czerwoną flagę.
- Maszynista drugiego pociągu zauważył dróżnika idącego z czerwoną flagą. Zatrzymał się w tunelu i zaczął cofać pociąg, aby upewnić się, że mógł do niego bezpiecznie wjechać.
- Pierwszy z dróżników chcąc poinformować kolegę na drugim końcu o sytuacji najpierw nadał sygnał ***pociąg w tunelu***, następnie chcąc się dowiedzieć czy oba pociągi już wyjechały zapytał sygnałem ***czy pociąg wyjechał z tunelu?***

SPECYFIKACJE NIEFORMALNE

Tymczasem na drugim końcu …

- Po opuszczeniu tunelu przez pierwszy pociąg, dróżnik wysłał sygnał ***tunel pusty,*** którym równocześnie odpowiedział na nieoczekiwane pytanie ***czy pociąg w tunelu***, które pojawiło się zaraz po powtórzonym sygnale ***pociąg w tunelu*** … i wpuścił kolejny pociąg ze swojej strony.
- Po chwili na cofający się pociąg najechał jadący z przeciwka …

Co zawiniło?

- Sygnalizacja?
- Człowiek?
- Czy źle zaprojektowany protokół?

Protokół InRes

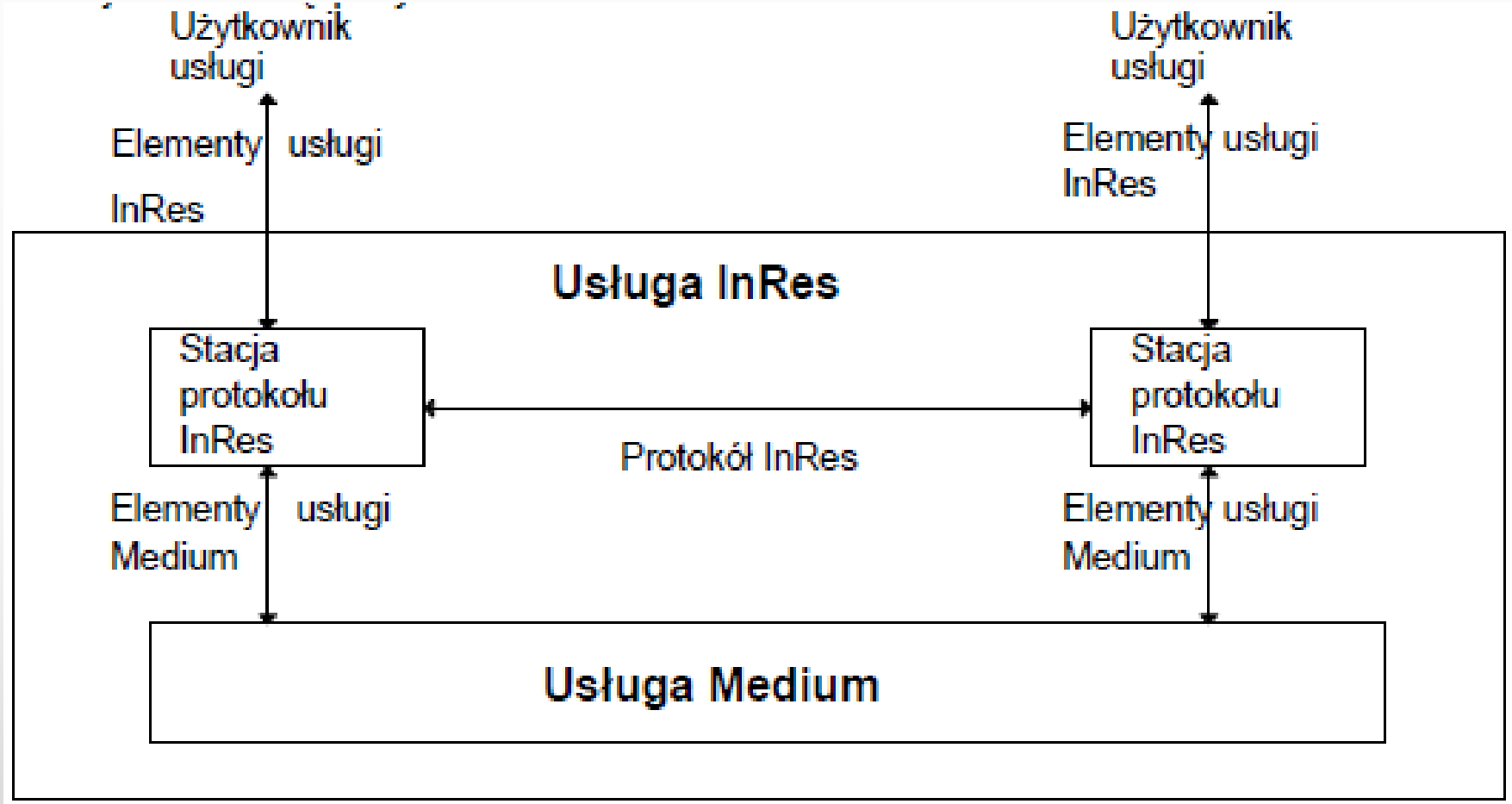
Zaczerpnięty z Hogrefe D.:Estelle, LOTOS und SDL, Springer Verlag

W przykładzie będą opisane dwie usługi i jeden protokół:

- **usługa Medium**, realizująca zawodne przesyłanie danych,
- **protokół InRes**, wykorzystujący usługę Medium, by zaoferować niezawodną usługę zorientowaną połączeniowo,
- **usługa InRes**, oferowana w oparciu o protokół InRes i usługę Medium.

Opisane usługi i protokół służą wyłącznie do ilustracji i nie można ich przypisywać żadnej z konkretnych warstw modelu OSI. Obie usługi oraz protokół są opisane werbalnie, bądź semiformalnie (harmonogramy).

SPECYFIKACJE NIEFORMALNE – PROTOKÓŁ INRES



SPECYFIKACJE NIEFORMALNE – PROTOKÓŁ INRES

W opisie wykorzystano następujące oznaczenia:

- SP - element usługi,
- SAP - punkt dostępu do usług,
- SDU - jednostka danych usługi.

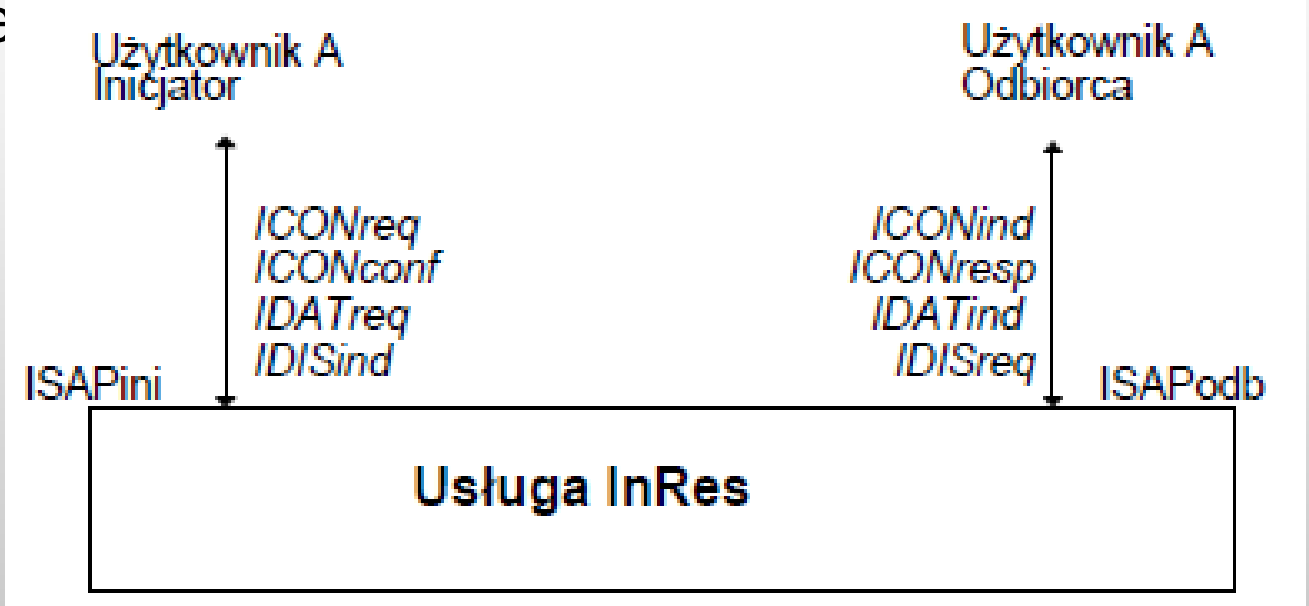
Jeśli którekolwiek z tych pojęć jest związane z usługą Medium, to jego oznaczenie poprzedza przedrostek M i tak MSDU oznacza jednostkę danych usługi Medium. Podobnie będzie dla protokołu InRes, gdzie zastosowano przedrostek I. Kolejność następnych punktów odpowiada kolejności opisywania zalecanej przy projektowaniu protokołów: najpierw rozważana jest usługa, która będzie oferowana przez protokół.

SPECYFIKACJE NIEFORMALNE – PROTOKÓŁ INRES

- ***Usługa InRes***

Przedstawiona usługa jest uproszczoną wersją usługi ABRACADABRA. **Usługa ta jest zorientowana połączeniowo.**

Oznacza to, że użytkownik A, który chciałby skomunikować się z użytkownikiem B za pośrednictwem tej usługi musi najpierw nawiązać połączenie, zanim zacznie wymieniać dane



SPECYFIKACJE NIEFORMALNE – PROTOKÓŁ INRES

Przyjęto, że usługa jest:

- asymetryczna i
- usługodawca posiada dwa SAPy.

Przez pierwszy SAP użytkownik A inicjuje połączenie, a potem przesyła dane. W drugim SAP-ie użytkownik B może zaakceptować albo odrzucić zgłoszenie nawiązania połączenia. W przypadku akceptacji poprzez ten właśnie SAP odbiera nadawane do niego dane.

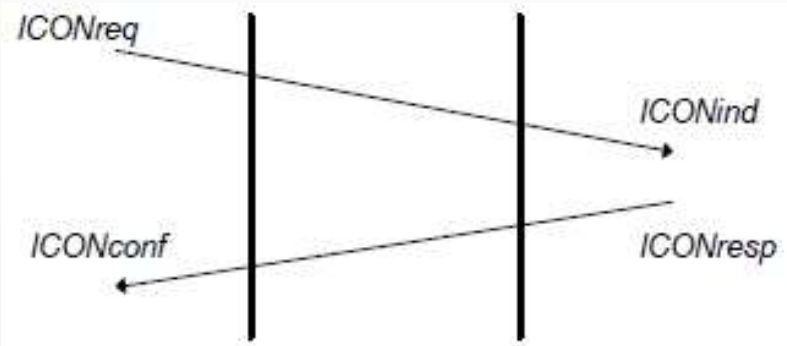
SPECYFIKACJE NIEFORMALNE – PROTOKÓŁ INRES

Komunikacja z usługodawcą realizowana jest za pomocą następujących elementów usług:

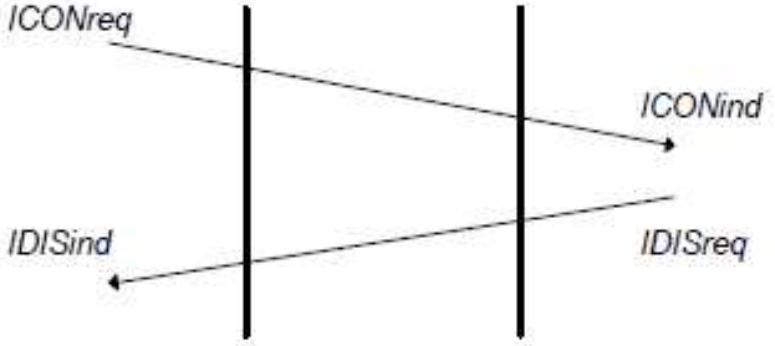
- **ICONreq:** żądanie nawiązania połączenia przez Inicjatora,
- **ICONind:** zgłoszenie nawiązania połączenia przez usługodawcę,
- **ICONresp**: odpowiedź Odbiorcy na zgłoszenie nawiązania połączenia,
- **ICONconf**: potwierdzenie nawiązania połączenia przez usługodawcę,
- **IDATreq(ISDU):** dane od Inicjatora do usługodawcy; parametrem tego SP jest ISDU,
- **IDATind(ISDU):** dane od usługodawcy do Odbiorcy; parametrem tego SP jest ISDU,
- **IDISreq**: żądanie rozłączenia połączenia przez Odbiorcę,
- **IDISind**: zgłoszenie rozłączenia przez usługodawcę.

SPECYFIKACJE NIEFORMALNE – PROTOKÓŁ INRES

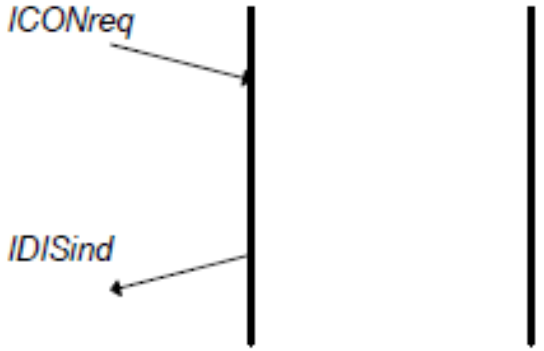
- Chronologię nadawania i odbioru elementów usług ilustrują (za pomocą harmonogramów) rysunki



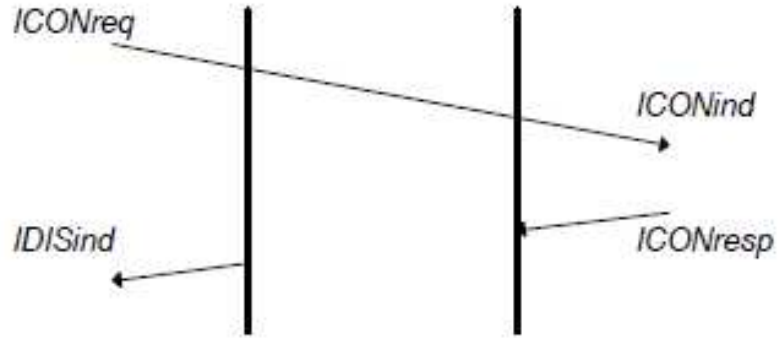
Rys 4 a) Udane nawiązanie połączenia



Rys 4 b) Nieudane nawiązanie połączenia
(odrzucenie przez Odbiorcę)

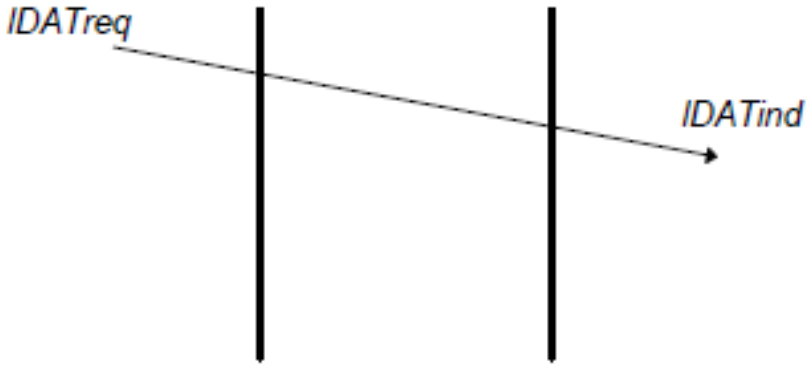


Rys 4 c) Nieudane nawiązanie połączenia
(Błędna transmisja żądania nawiązania połączenia)



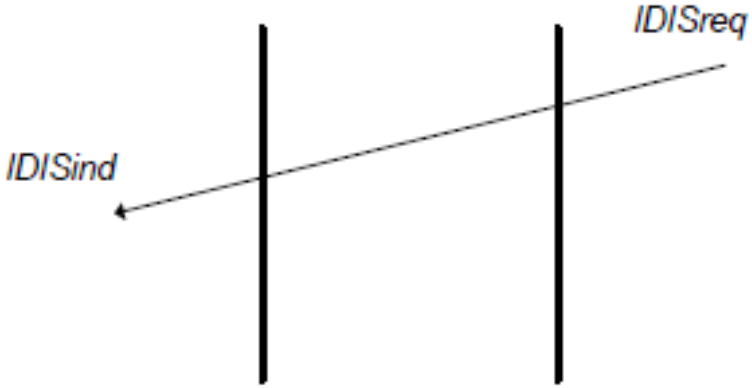
Rys 4 d) Nieudane nawiązanie połączenia
(Błędna transmisja odpowiedzi)

SPECYFIKACJE NIEFORMALNE – PROTOKÓŁ INRES

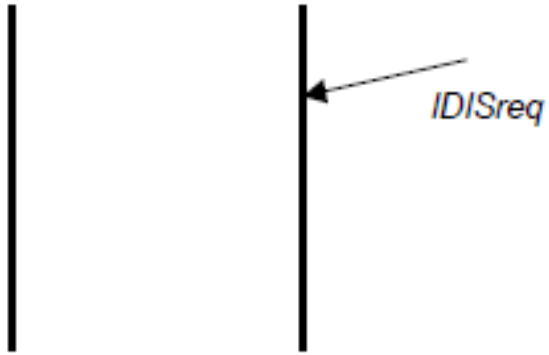


Rys 4 e) Udane nadanie danych

Rys 4 f) Nieudane nadanie danych
(Błędna transmisja danych)



Rys 4 g) Udane rozłączenie połączenia



Rys. 4 h) Nieudane rozłączenie połączenia
(Błędna transmisja żądania rozłączenia)

SPECYFIKACJE NIEFORMALNE – PROTOKÓŁ INRES

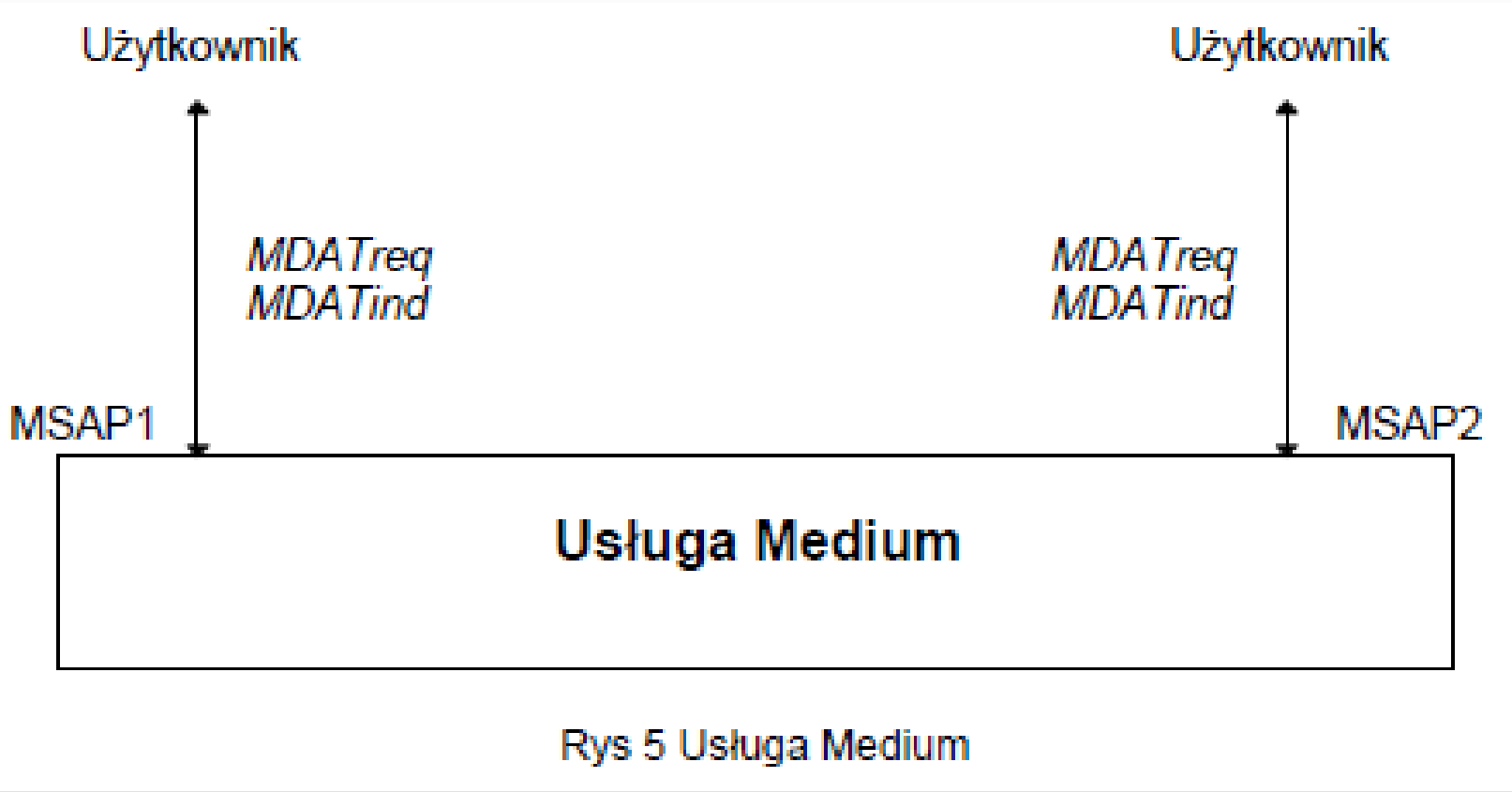
Usługa Medium

Usługa Medium posiada dwa SAPy: MSAP1 i MSAP2. Jest symetryczna, bezpołączeniowa i może być używana poprzez elementy MDATreq i MDATind, z których każdy posiada parametr MSDU.

Za pomocą tych elementów dane (MSDU) mogą być przekazywane pomiędzy SAPami. Użytkownik usługi przekazuje dane usłudze korzystając z MDATreq, a usługa otrzymane dane przekazuje innemu użytkownikowi korzystając z MDATind.

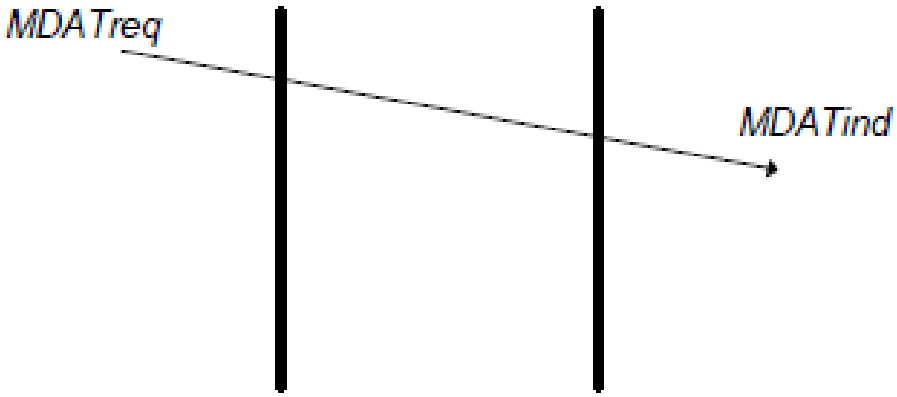
Przekazanie danych nie jest bezbłędne w tym sensie, że dane mogą ulec zaginięciu. Nie mogą jednak pojawić się przekłamania w transmisji, ani zwielokrotnienia danych, ani wygenerowanie danych.

SPECYFIKACJE NIEFORMALNE – PROTOKÓŁ INRES

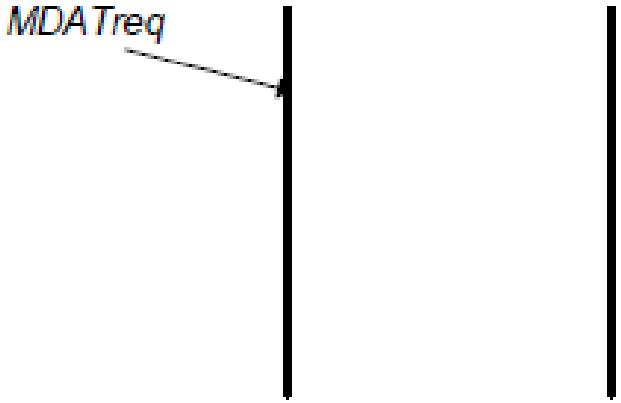


Rys 5 Usługa Medium

SPECYFIKACJE NIEFORMALNE – PROTOKÓŁ INRES

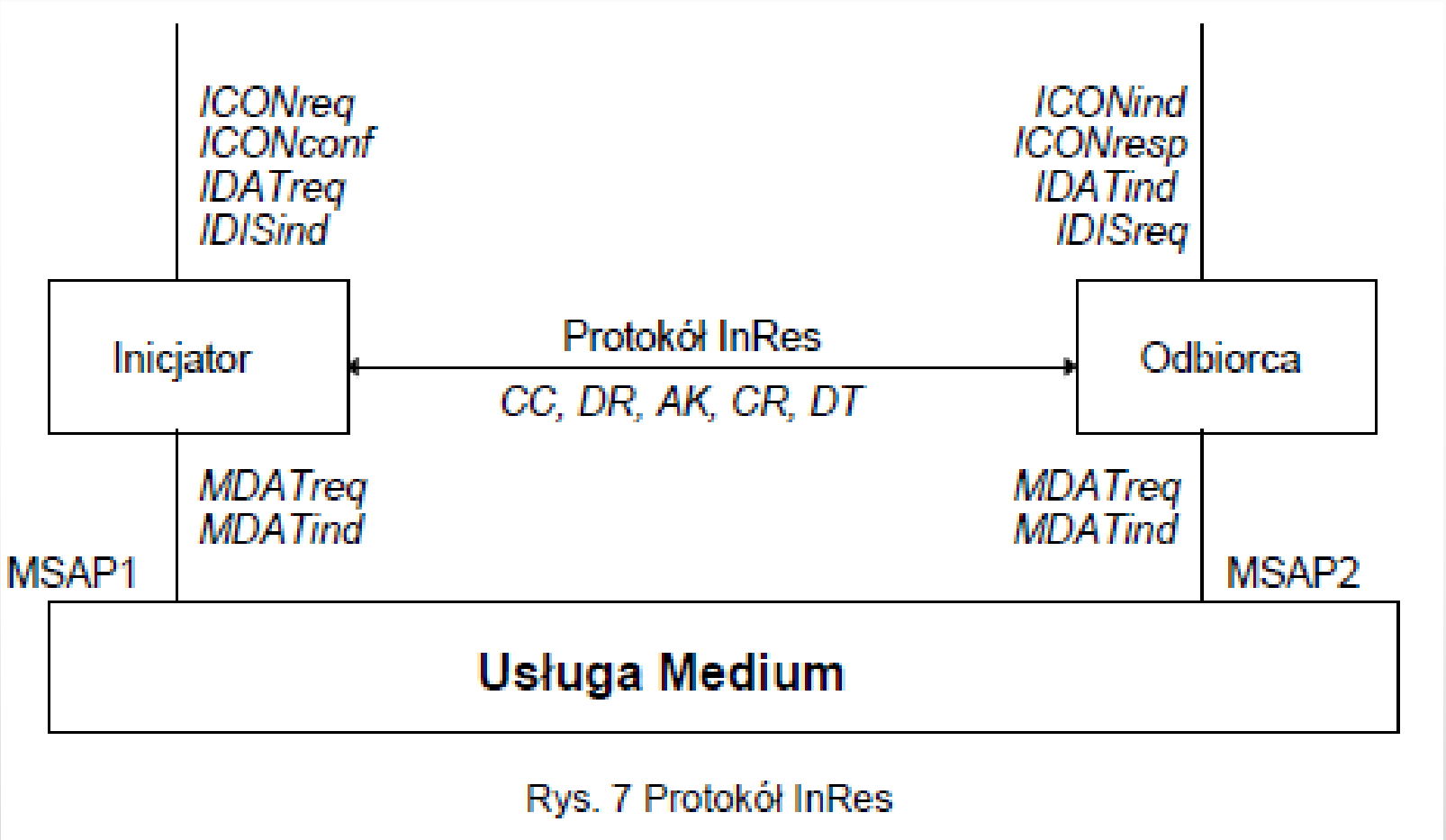


Rys 6 a) Udane nadanie danych



Rys. 6 b) Nieudane nadanie danych
(Błędna transmisja danych)

SPECYFIKACJE NIEFORMALNE – PROTOKÓŁ INRES



Rys. 7 Protokół InRes

SPECYFIKACJE NIEFORMALNE – PROTOKÓŁ INRES

Protokół InRes

Protokół korzysta z bardzo prostej, lecz zawodnej usługi Medium i oferuje usługę InRes.

Ogólne własności protokołu

Protokół jest protokołem zorientowanym połączeniowo i działa między dwiema stacjami protokołu Inicjator i Odbiorca. Stacje protokołu komunikują się dzięki wymianie jednostek danych protokołu CR, CC, DT, AK i DR. Znaczenie i parametry tych jednostek danych są opisane w tabeli 1.

Komunikacja między stacjami protokołu odbywa się w trzech fazach: faza nawiązania połączenia, faza nadawania danych, faza rozłączenia połączenia. W każdej z trzech faz znaczenie mają tylko określone PDU i SP. PDU i SP nieoczekiwane przez Inicjatora i Odbiorcę są ignorowane.

SPECYFIKACJE NIEFORMALNE – PROTOKÓŁ INRES

Tablica 1 Jednostki danych protokołu InRes

PDU	Znaczenie	Parametry	Odpowiadające SP
CR	nawiązanie połączenia	brak	ICONreq, ICONind
CC	potwierdzenie połączenia	brak	ICONresp, ICONconf
DT	transmisja danych	nr kolejny, jednostka danych, usługi (ISDU)	IDATreq, IDATind
AK	potwierdzenie	nr kolejny	-
DR	rozłączenie połączenia	brak	IDISreq, IDISind

SPECYFIKACJE NIEFORMALNE – PROTOKÓŁ INRES

- *Faza nawiązania połączenia*

Nawiązanie połączenia jest inicjowane przez stację protokołu Inicjatora elementem ICONreq. Stacja Inicjator wysyła CR do stacji Odbiorca. Odbiorca odpowiada CC lub DR. W przypadku otrzymania CC Inicjator nadaje ICONconf do użytkownika i rozpoczyna się faza nadawania danych. Jeśli Inicjator otrzyma od Odbiorcy DR, to rozpoczyna się faza rozłączenia połączenia; gdy Inicjator w ciągu 5 sekund nie otrzyma odpowiedzi to ponownie wysyła CR. Jeśli po czterech ponowieniach nadal nie otrzyma odpowiedzi, to przechodzi do fazy rozłączenia połączenia. Gdy Odbiorca otrzyma od Inicjatora CR, to wysyła ICONind do użytkownika, z którym współpracuje. Użytkownik ten może odpowiedzieć ICONresp lub IDISreq. ICONresp jest potwierdzeniem nawiązania połączenia, Odbiorca nadaje CC do Inicjatora i rozpoczyna się faza nadawania danych. Po otrzymaniu IDISreq Odbiorca zaczyna fazę rozłączenia.

SPECYFIKACJE NIEFORMALNE – PROTOKÓŁ INRES

- *Faza nadawania danych*

Jeśli użytkownik przekaże stacji Inicjator element IDATreq, to Inicjator nadaje DT do Odbiorcy i może odebrać następny IDATreq od użytkownika. IDATreq zawiera parametr w postaci jednostki danych usługi ISDU, za pomocą, której użytkownik A przekazuje informacje użytkownikowi B. Informacje użytkownika A zawarte w IDATreq będą bez zmian przekazane jako jednostka danych protokołu DT. Po nadaniu DT Inicjator oczekuje 5 sekund na AK poprawne potwierdzenie odbioru od Odbiorcy. Po tym czasie DT jest ponownie wysyłane. Po czterech nieudanych próbach nadania DT Inicjator przechodzi do fazy rozłączenia połączenia.

SPECYFIKACJE NIEFORMALNE – PROTOKÓŁ INRES

Zarówno DT, jak i AK posiadają parametr w postaci binarnego numeru kolejnego (0 albo 1). Po nawiązaniu połączenia Inicjator rozpoczyna transmisję od DT z numerem 1. Poprawne potwierdzenie AK zawiera zawsze ten sam numer kolejny, który miało potwierdzane DT. Po odebraniu poprawnego AK następne DT jest wysyłane z następnym (a więc innym) numerem kolejnym. Jeśli Inicjator odbierze AK z niewłaściwym numerem kolejnym, to nadaje ostatnie DT jeszcze raz. DT jest powtarzane również wtedy, gdy po 5 sekundach nie jest odebrane AK. Jedno DT może być w sumie cztery razy powtórzone. Po czterech nieskutecznych powtórzeniach Inicjator rozpoczyna fazę rozłączenia połączenia. To samo dzieje się po odebraniu DR.

SPECYFIKACJE NIEFORMALNE – PROTOKÓŁ INRES

Po nawiązaniu połączenia Odbiorca oczekuje najpierw na DT z numerem kolejnym 1. Po odebraniu DT z oczekiwanym numerem kolejnym Odbiorca przekazuje dane (ISDU) do swojego użytkownika oraz nadaje potwierdzenie AK z identycznym numerem kolejnym do Inicjatora. DT z numerem kolejnym innym niż oczekiwany jest potwierdzane przez AK z ostatnim poprawnie odebranym numerem kolejnym. Dane użytkownika (ISDU) niewłaściwego DT są ignorowane. Jeśli Odbiorca otrzyma CR, to przechodzi do fazy nawiązania połączenia. Przy IDISreq Odbiorca rozpoczyna fazę rozłączenia połączenia.

- *Faza rozłączenia połączenia*

IDISreq od użytkownika prowadzi do nadania przez Odbiorcę DR do Inicjatora. Potem Odbiorca może ponownie przyjąć zgłoszenie nawiązania połączenia CR od Inicjatora. Otrzymanie DR przez Inicjatora powoduje nadanie IDISind do użytkownika. IDISind jest wysyłany do użytkownika, gdy CR albo DT był czterokrotnie nadawany bez powodzenia. Po tym można zacząć nawiązywanie nowego połączenia.

SDL

SPECIFICATION AND DESCRIPTION LANGUAGE

Formalne modele protokołów i usług

Model procesu komunikacyjnego: rozszerzony automat skończony

Najczęściej przyjmowanym formalnym modelem procesu komunikacyjnego jest model rozszerzonego automatu skończonego. Proces komunikacyjny opisuje się jako złożenie przejść miedzy stanami procesu, które można opisać dwoma funkcjami:

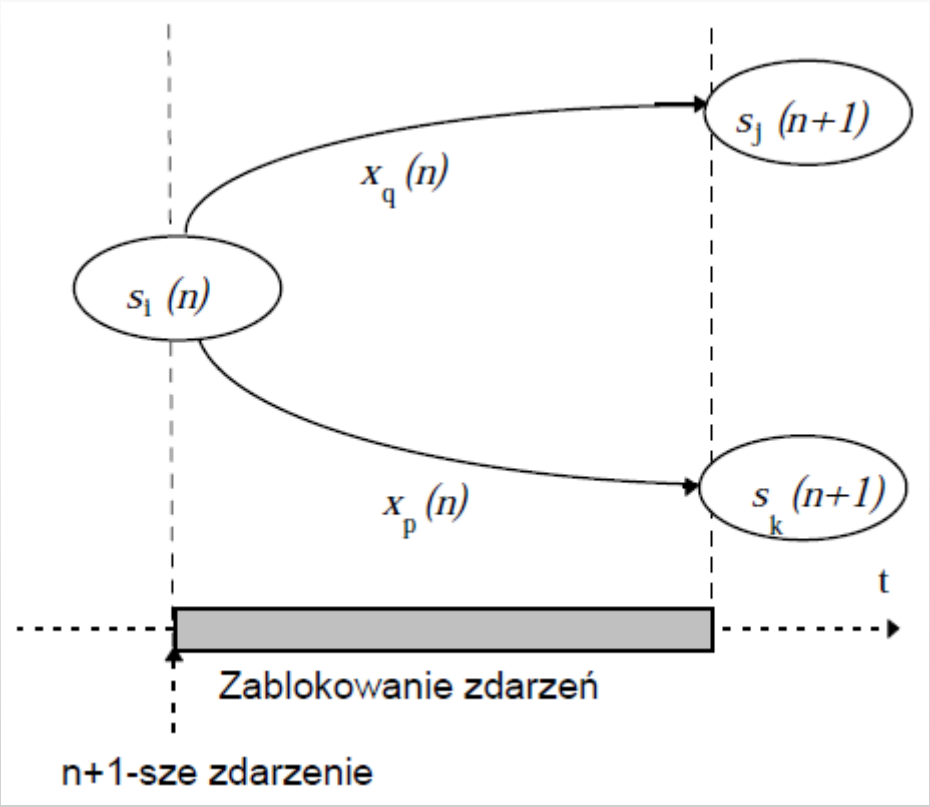
$$\begin{cases} s(n+) = f(s(n), x(n)) \text{ - funkcja przejscia} \\ y(n) = g(s(n), x(n)) \text{ - funkcja wyjscia} \end{cases}$$

gdzie:

- *s(n)* - stan procesu po *n*-tym zdarzeniu
- *x(n)* - *n*-te zdarzenie (sygnał wejściowy, pobudzenie)
- *y(n)* - reakcja na *n*-te zdarzenie (sygnały wyjściowe).

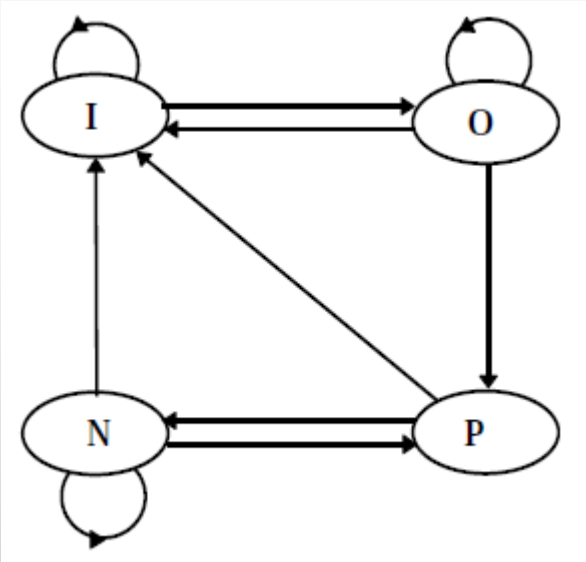
SPECYFICATION AND DESCRIPTION LANGUAGE

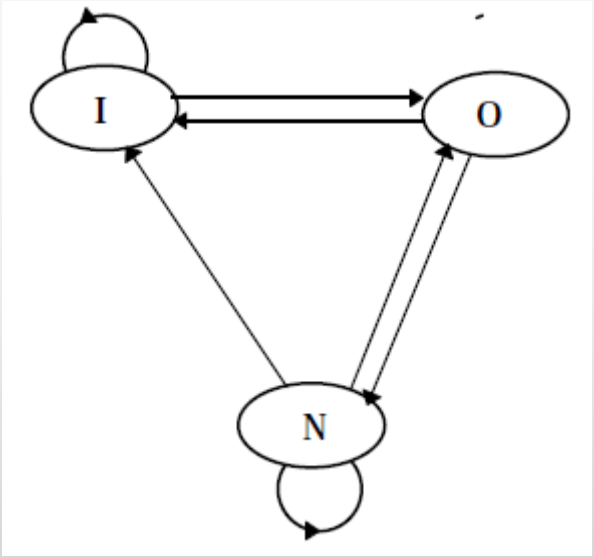
Poniższy rysunek ilustruje funkcje przejścia procesu. Nowy stan procesu po zajściu zdarzenia zależy tylko i wyłącznie od rodzaju zdarzenia *x* i stanu procesu w chwili zajścia tego zdarzenia.



SPECYFICATION AND DESCRIPTION LANGUAGE

Procesy komunikacyjne mają strukturę, która można modelować przy użyciu **automatu skończonego**. Stany automatu są związane z oczekiwaniem na **zdarzenia (komunikaty)** wytwarzane przez współpracujące procesy lub otoczenie systemu. Reakcje automatu są również komunikatami.





Rysunki przedstawiają grafy przejść dla procesów inicjującego połączenie i odbierającego w przykładzie podanym w poprzednim rozdziale.

SPECIFICATION AND DESCRIPTION LANGUAGE

SDL (Specification and Description Language) jest najstarszą metodą formalnej specyfikacji protokołów. Opracowano ją pod koniec lat sześćdziesiątych do opisu protokołów połączeń telefonicznych. Metoda ta jest objęta Zaleceniami CCITT (ITU) od roku 1976.

Ze względu na to, że SDL był opracowany do opisu protokołów połączeń telefonicznych, miał być zrozumiały przede wszystkim dla użytkowników systemów telefonicznych. Spowodowało to istnienie dwóch równorzędnych reprezentacji specyfikacji **graficznej i tekstowej.** U podstaw obu tych reprezentacji leży oczywiście ten sam model formalny systemu. W dalszej części będziemy omawiali obie reprezentacje języka SDL jednocześnie.

Elementy języka

Zasadniczym elementem języka jest **proces**. Procesy są środkami reprezentacji dynamicznego zachowania się systemu. Zakłada się, że procesy wykonywane są współbieżnie. Procesy oczekują w dyskretnych **stanach** na **sygnały** ze swego otoczenia. Proces działa tylko w odpowiedzi na zewnętrzne dyskretne **pobudzenia** i wynikiem jego działania jest wysłanie dyskretnej odpowiedzi do otoczenia. Po odebraniu sygnału pobudzenia proces dokonuje zmiany stanu (tranzycji). Podczas zmiany stanu proces realizuje akcje w trakcie, których przetwarza odebraną i lokalną informacje oraz wysyła **sygnały** (**odpowiedzi**) do innych procesów i do otoczenia systemu. Odpowiedz ta jest odbierana przez określone procesy jako pobudzenie.

SPECIFICATION AND DESCRIPTION LANGUAGE

Podstawowe elementy języka SDL to:

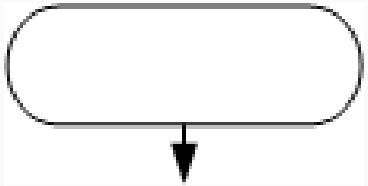
- **stan procesu (STATE)**
- **sygnał pobudzenia (INPUT)**
- **sygnał reakcji (OUTPUT)**
- **sygnał upływu czasu (TIMER)**

Szczegółową specyfikację SDL zawiera dokument Z100.pdf

SPECIFICATION AND DESCRIPTION LANGUAGE

Początek procesu

START nazwa_stanu;



Bieżący stan procesu

STATE nazwa_stanu;

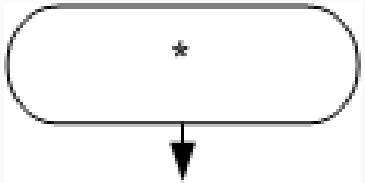
Następny stan procesu:

NEXTSTATE nazwa_stanu;

SPECIFICATION AND DESCRIPTION LANGUAGE

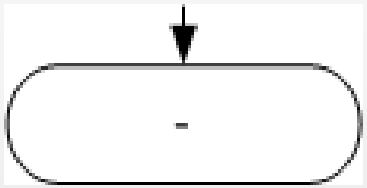
Konwencja gwiazdkowa – każdy stan procesu:

STATE *;



Konwencja kreskowa – powrót do stanu początkującego akcję:

NEXTSTATE -;



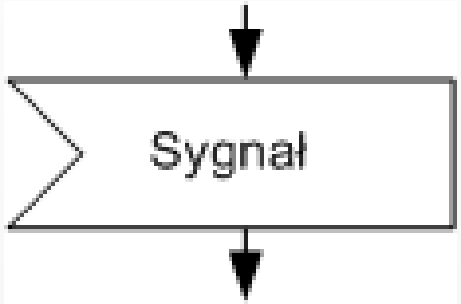
Komentarz:

/* komentarz */

SPECIFICATION AND DESCRIPTION LANGUAGE

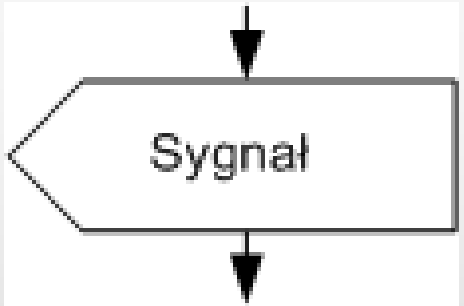
Sygnał wejściowy (pobudzenie):

INPUT nazwa;



Sygnał wyjściowy (reakcja):

OUTPUT nazwa;



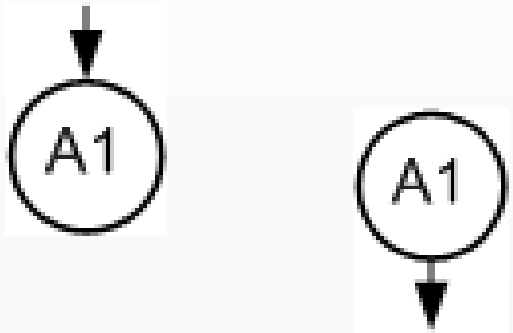
SPECIFICATION AND DESCRIPTION LANGUAGE

Łączniki i etykiety:

JOIN A1;

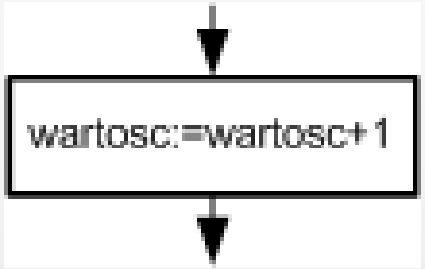
...

A1:



Zadanie:

wartosc := wartosc + 1;



SPECIFICATION AND DESCRIPTION LANGUAGE

Decyzja:

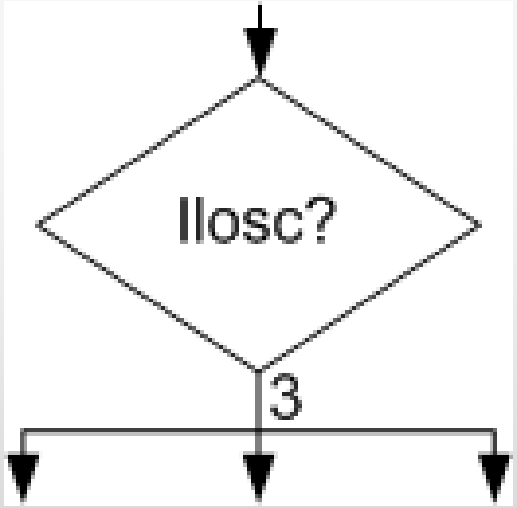
DECISION warunek;

(wariant1): zadanie1;

(wariant2): zadanie2;

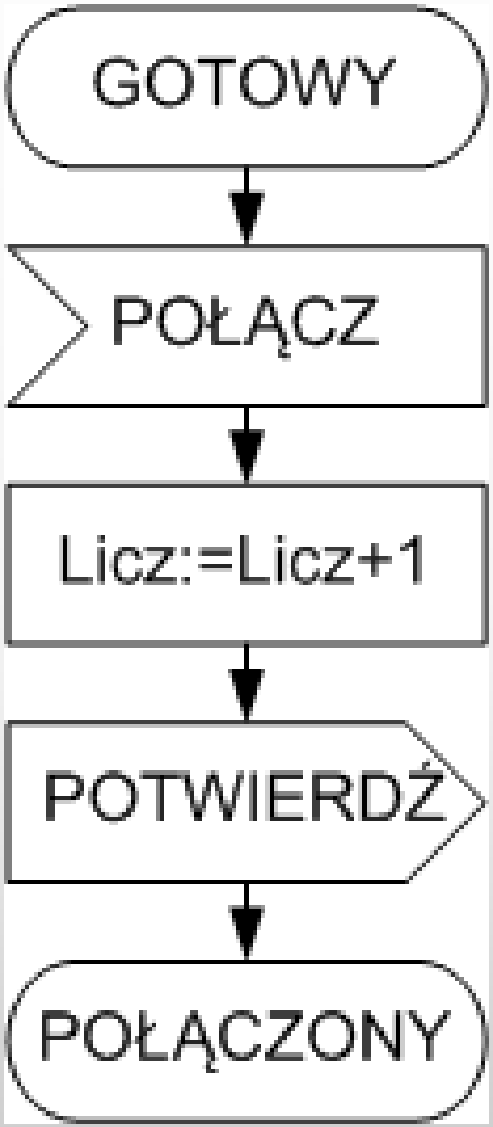
ELSE: zadanie3;

ENDDECISION;



SPECIFICATION AND DESCRIPTION LANGUAGE

Przykład:



SPECIFICATION AND DESCRIPTION LANGUAGE

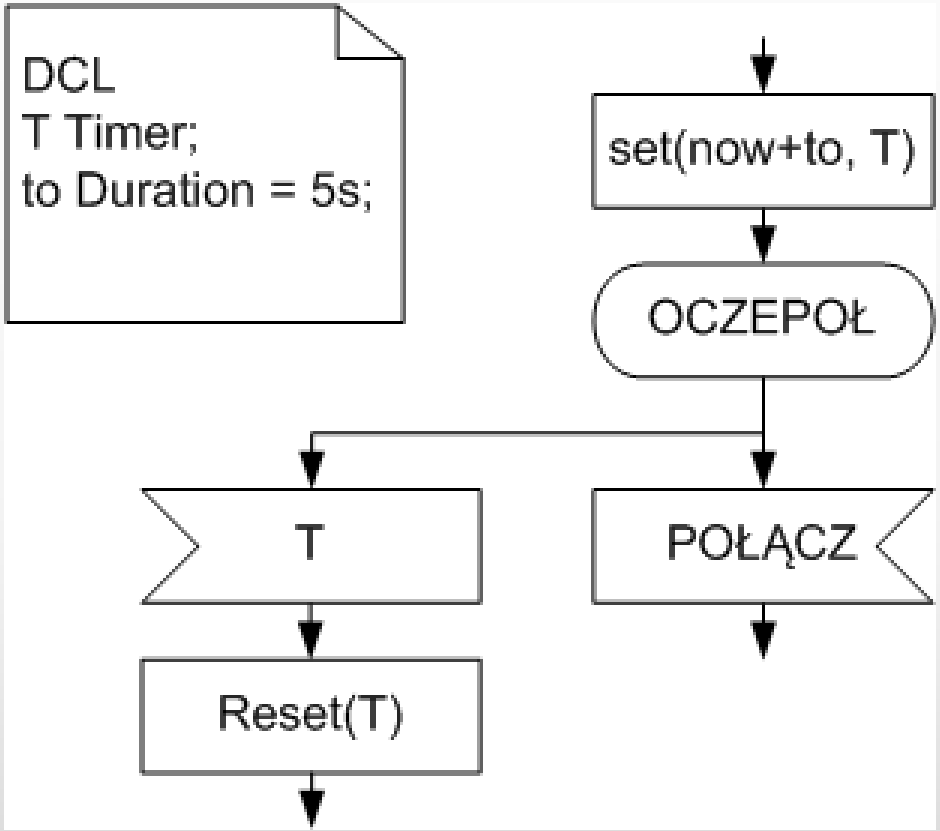
Pomiar czasu

Wszystkie procesy w systemie mają dostęp do zegarów (zmiennych typu Timer) odmierzających czas bezwzględny. Wartości czasu można przetwarzać w zmiennych typu Time, przedziały czasu w zmiennych typu Duration.

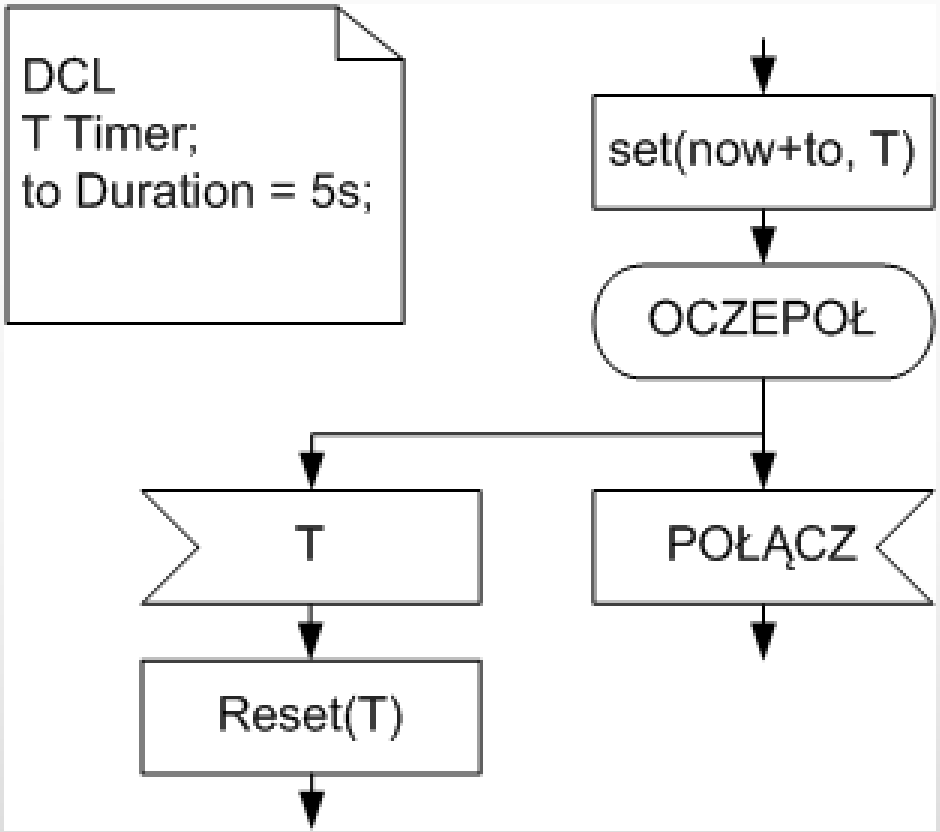
- **TIMER** – typ zegara
- **SET** – funkcja ustawiająca zegar
- **RESET** – funkcja zerująca zegar
- **NOW** – zmienna przechowująca bieżący czas
- **TIME** – typ bezwzględnej chwili czasu
- **DURATION** – typ przedziału czasu

Dla osiągnięcia odpowiedniego poziomu abstrakcji opisu w języku SDL nie ma zdefiniowanej jednostki czasu.

SPECIFICATION AND DESCRIPTION LANGUAGE



SPECIFICATION AND DESCRIPTION LANGUAGE



SPECIFICATION AND DESCRIPTION LANGUAGE

Predefiniowane typy zmiennych:

- **INTEGER** - typ liczb całkowitych
- **NATURAL** - typ liczb naturalnych
- **BOOLEAN** - typ wartości logicznych
- **CHARACTER** - typ znaku
- **REAL** - typ liczb rzeczywistych
- **PID** - typ identyfikatora procesu
- **DURATION** - typ przedziału czasu
- **TIMER** - typ zegara
- **TIME** - typ chwili czasu
- **CHARSTRING** - typ łańcucha znakowego
- **ARRAY** - typ macierzowy
- **POWERSET** - zbiór podzbiorów

SPECIFICATION AND DESCRIPTION LANGUAGE

Procesy mogą być dynamicznie kreowane i niszczone zarówno w czasie inicjacji systemu jak i w czasie jego życia. Proces nie może być zniszczony przez inny proces, lecz może sam się unicestwić.

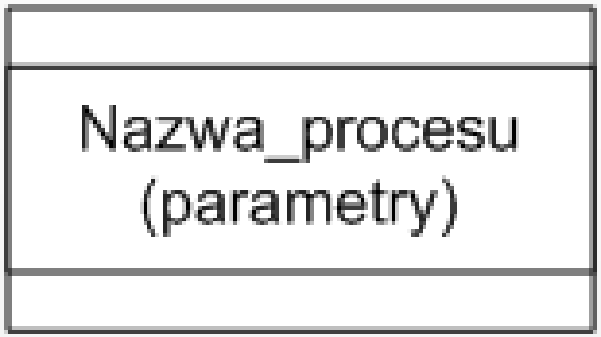
Zmienne predefiniowane związane z procesami:

- **PARENT** – PID rodzica - procesu kreującego
- **SELF** – własny PID
- **OFFSPRING** – PID ostatnio utworzonego potomka
- **SENDER** – PID nadawcy odebranego sygnału

SPECIFICATION AND DESCRIPTION LANGUAGE

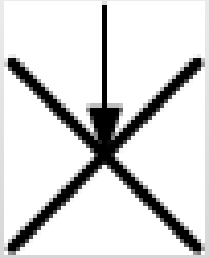
Utworzenie procesu:

CREATE Nazwa_procesu(parametry);



Zakończenie procesu:

STOP;



SPECIFICATION AND DESCRIPTION LANGUAGE

Instancja procesu:

PROCESS Nazwa(n1,n2);

FPAR…;

…

ENDPROCESS;

Gdzie:

- n1 - liczba instancji procesu aktywna w miejscu użycia symbolu;
- n2 - maksymalna liczba instancji tego procesu w całej specyfikacji.

Wartości domyślne obu parametrów to: n1 = 1, n2 = oo

SPECIFICATION AND DESCRIPTION LANGUAGE

Komunikacja procesów

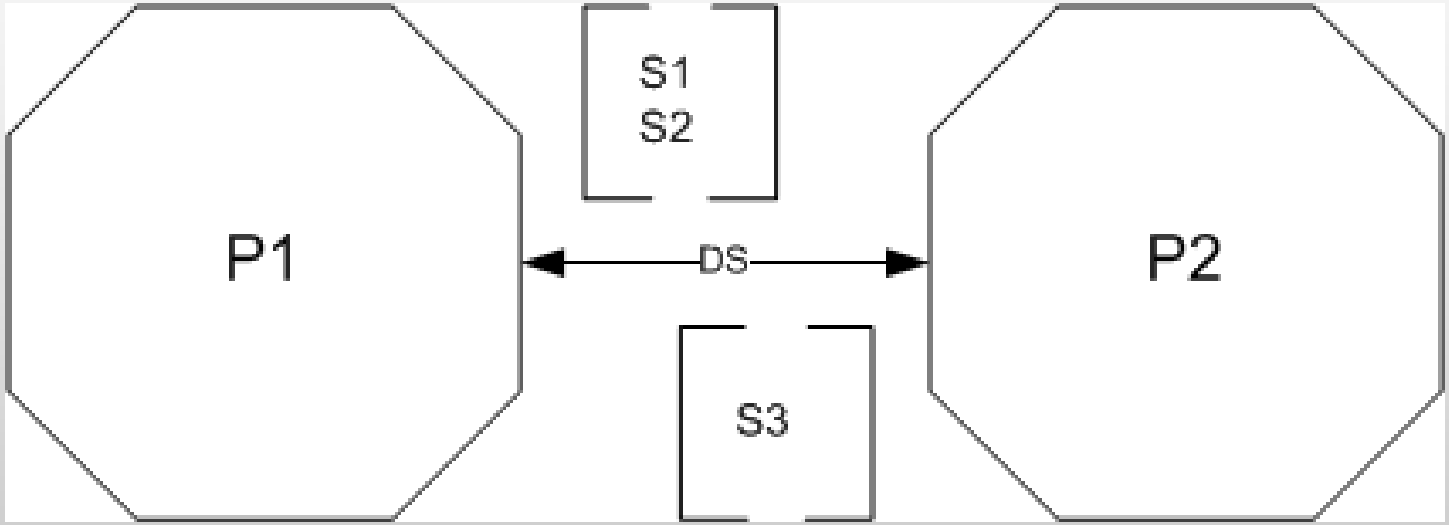
SIGNAL

S1, S2, S3;

SIGNALROUTE DS

FROM P1 TO P2 WITH S3

FROM P2 TO P1 WITH S1, S2;



Adresowanie sygnałów

Adresowanie bezpośrednie do procesu

OUTPUT A TO SENDER;

OUTPUT A TO OFFSPRING;

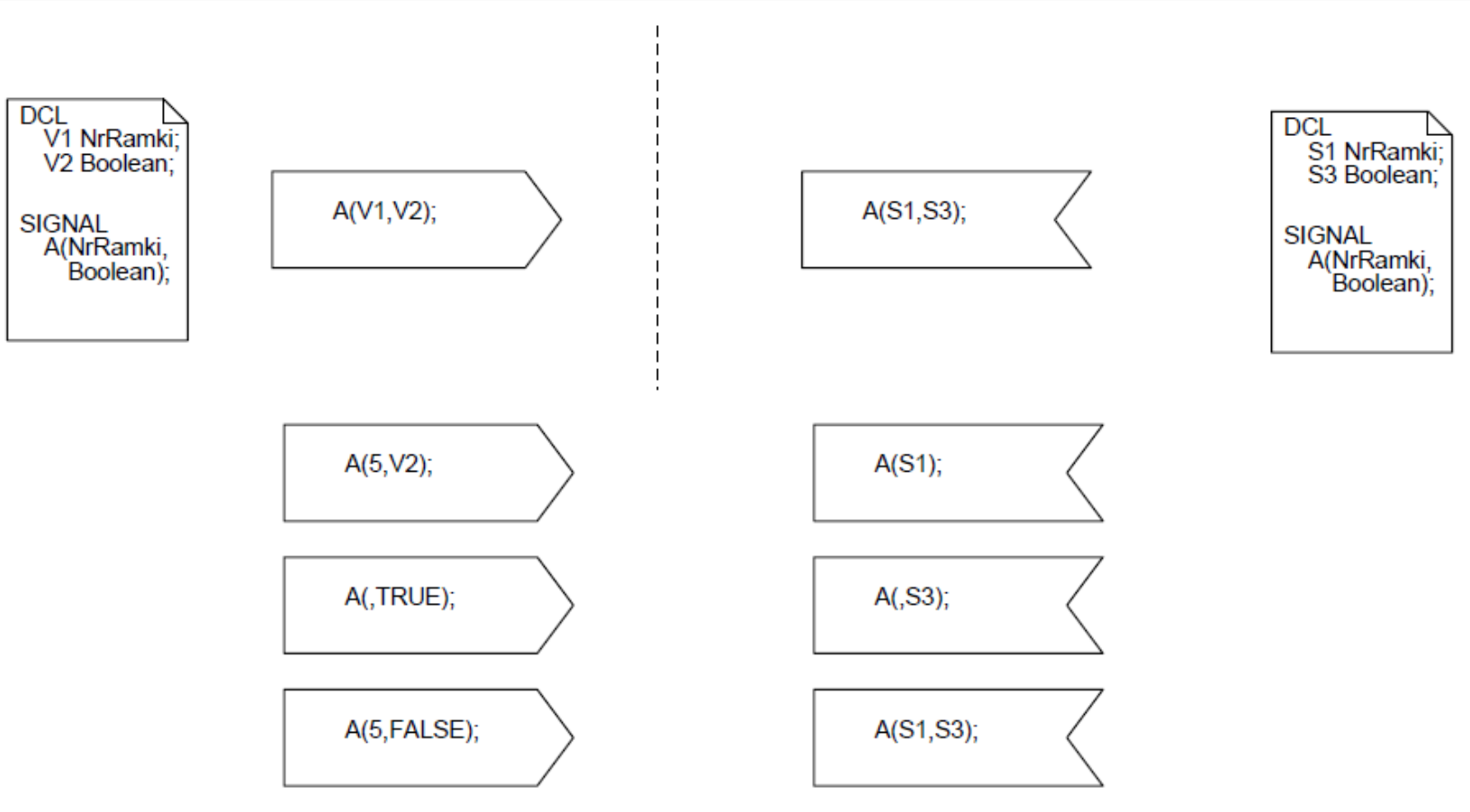
OUTPUT A TO P1;

Adresowanie pośrednie za pomocą drogi sygnałowej, kanału albo bramki:

OUTPUT A VIA DS;

OUTPUT A VIA GATE1;

Przekazywanie wartości przez sygnały

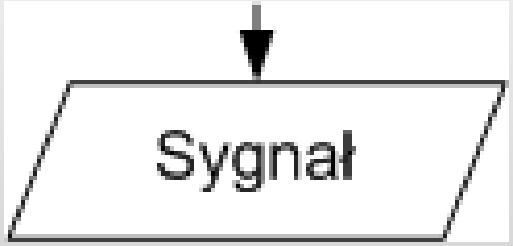


Semantyka komunikacji procesów

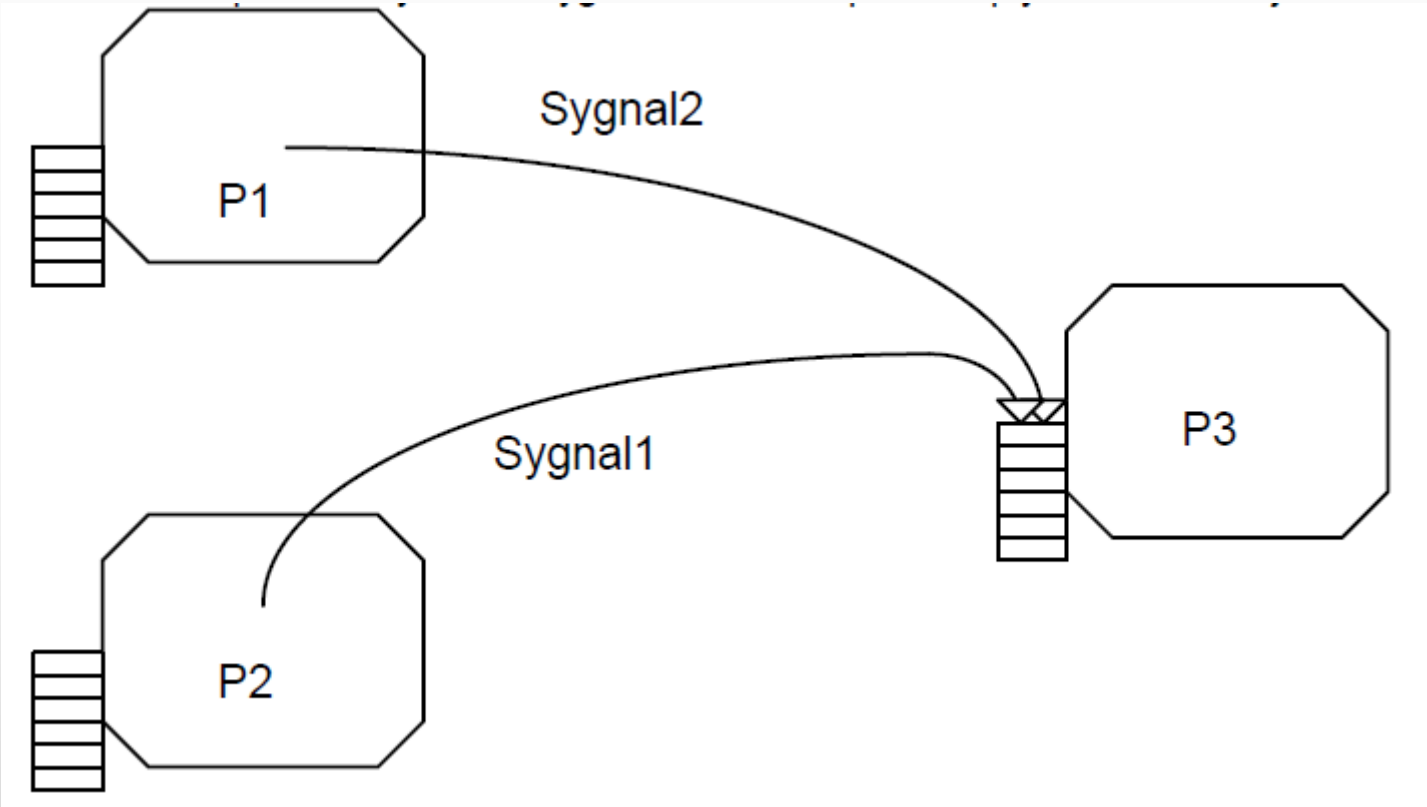
Na odbiór w procesie może oczekiwać kilka sygnałów. Oczekują one na obsługę w kolejce portu wejściowego. Kolejka ta jest zasadniczo kolejką typu FIFO "Pierwszy Zgłoszony Pierwszy Obsłużony", lecz proces ma możliwość "przechowywania" sygnałów i w ten sposób wpływania na kolejność ich obsługi.

Zachowaj sygnał:

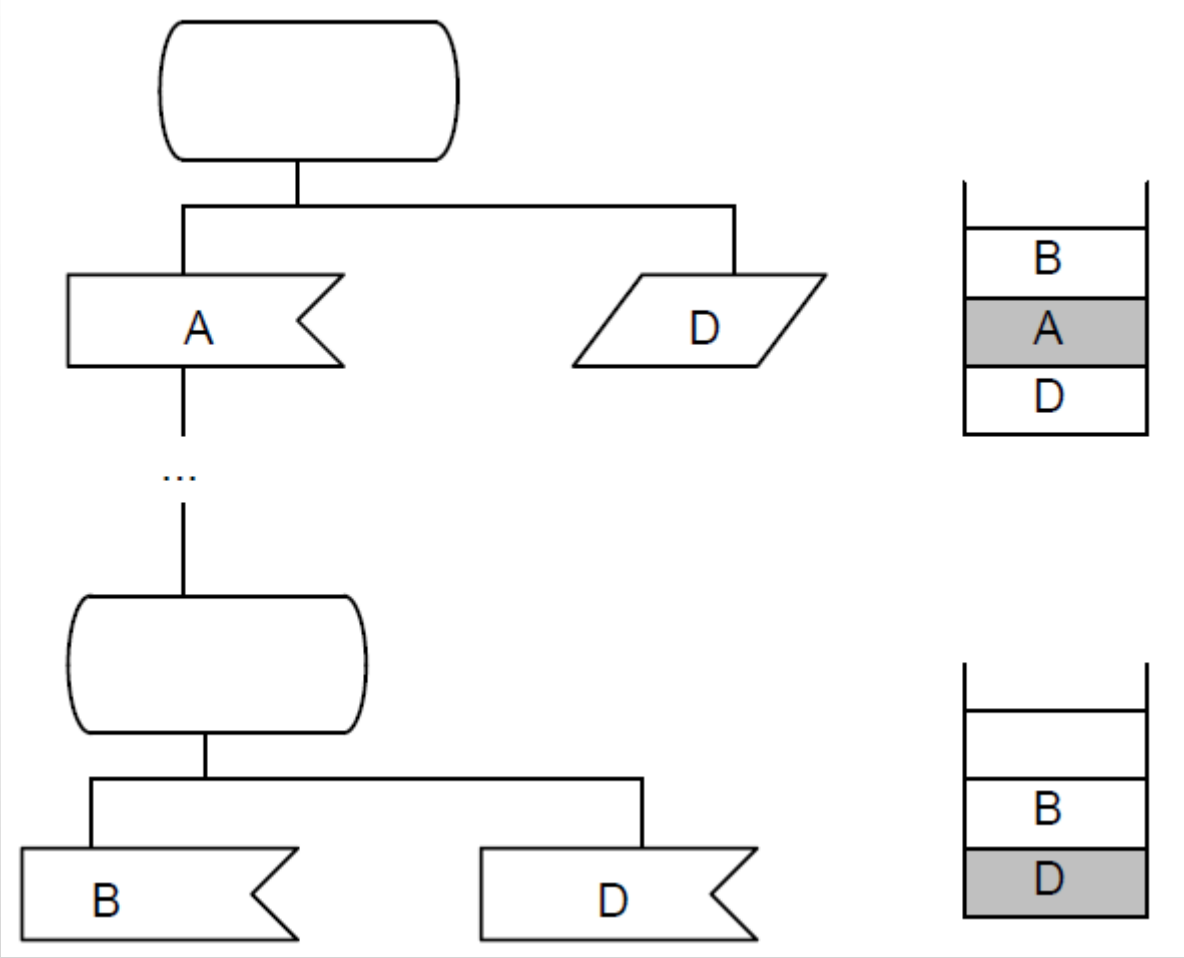
SAVE nazwa_sygnału;



SPECIFICATION AND DESCRIPTION LANGUAGE



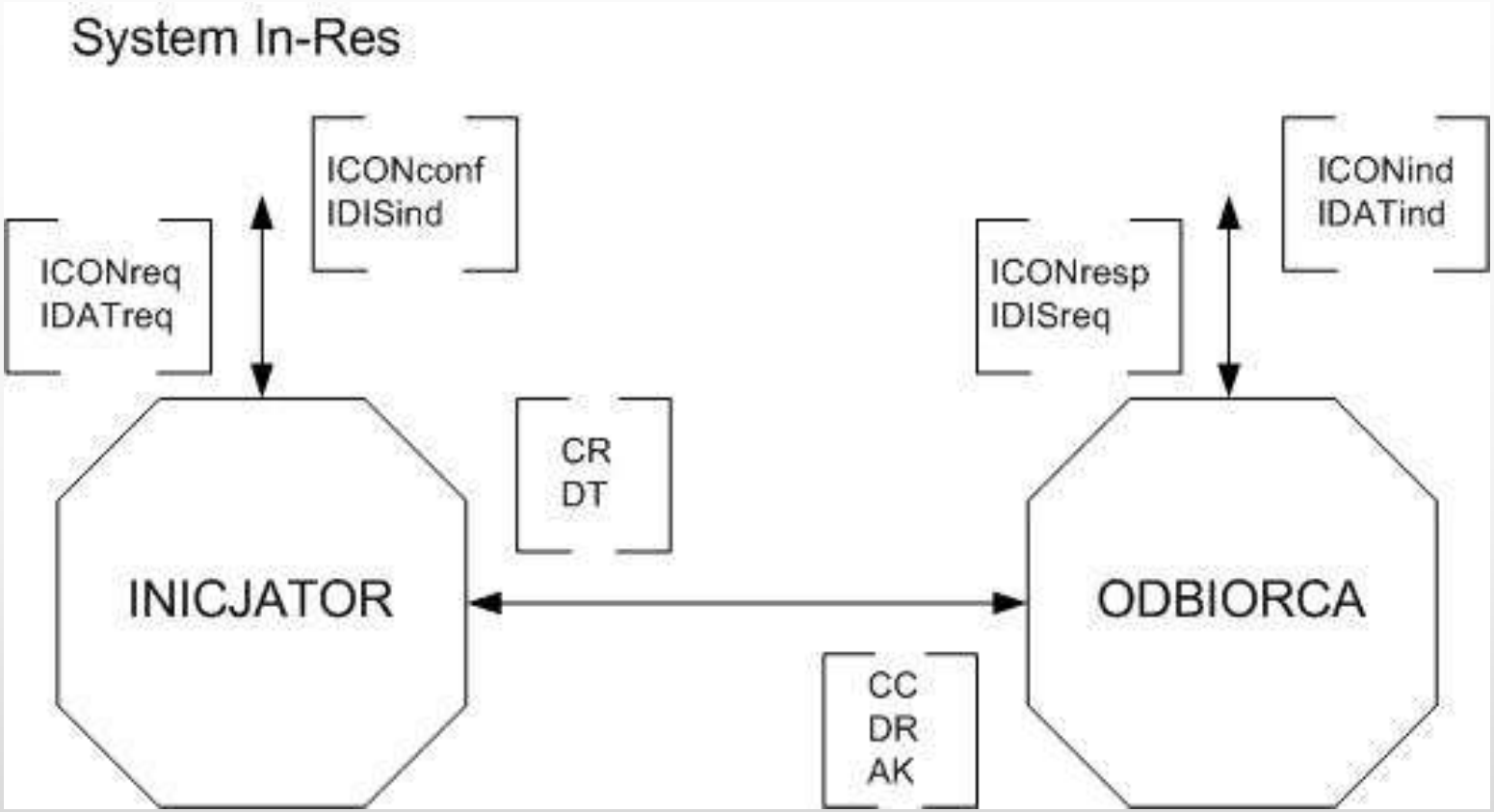
SPECIFICATION AND DESCRIPTION LANGUAGE



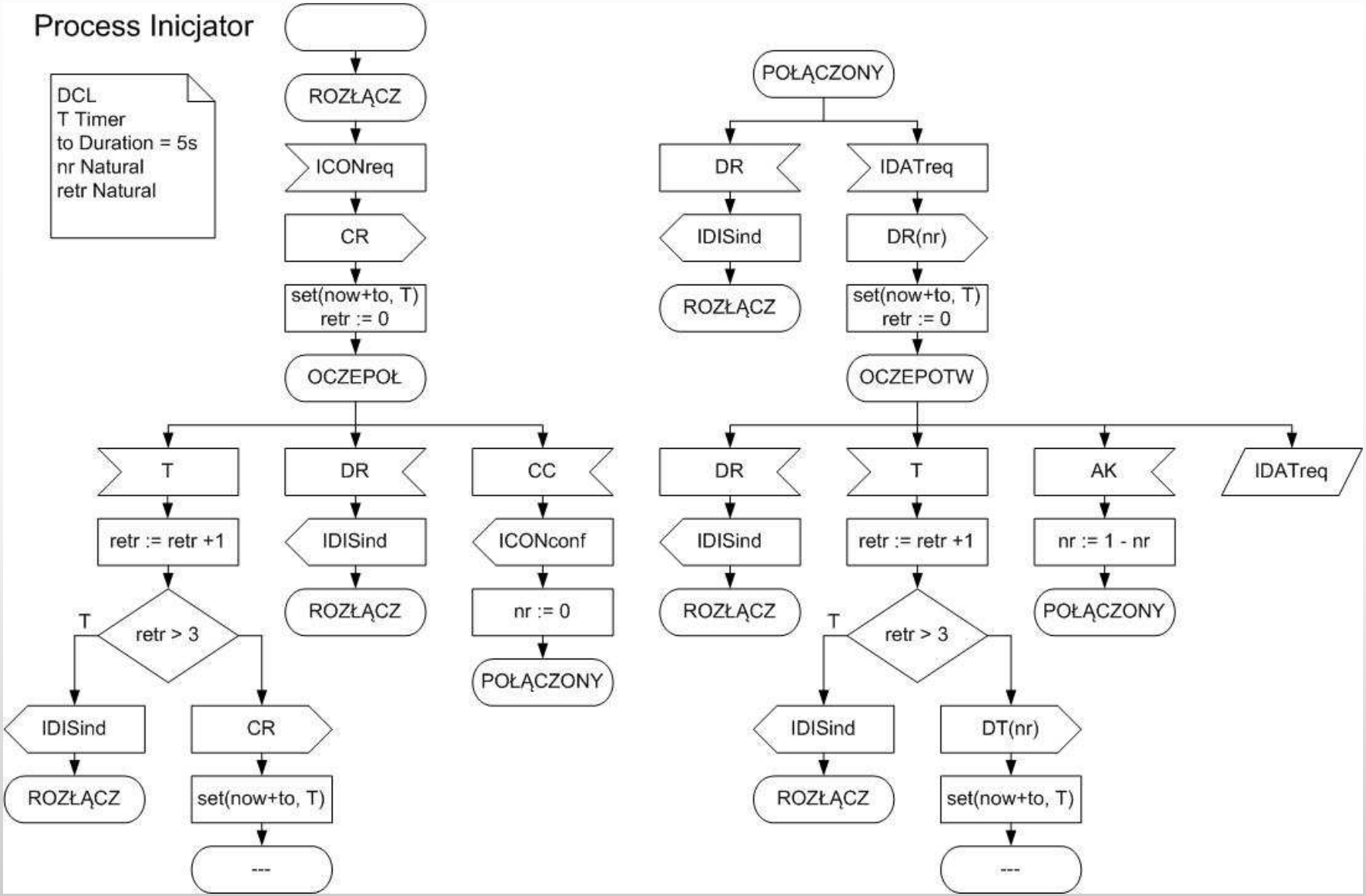
PRZYKŁADY SPECYFIKACJI FORMALNEJ

SPECYFIKACJE FORMALNE

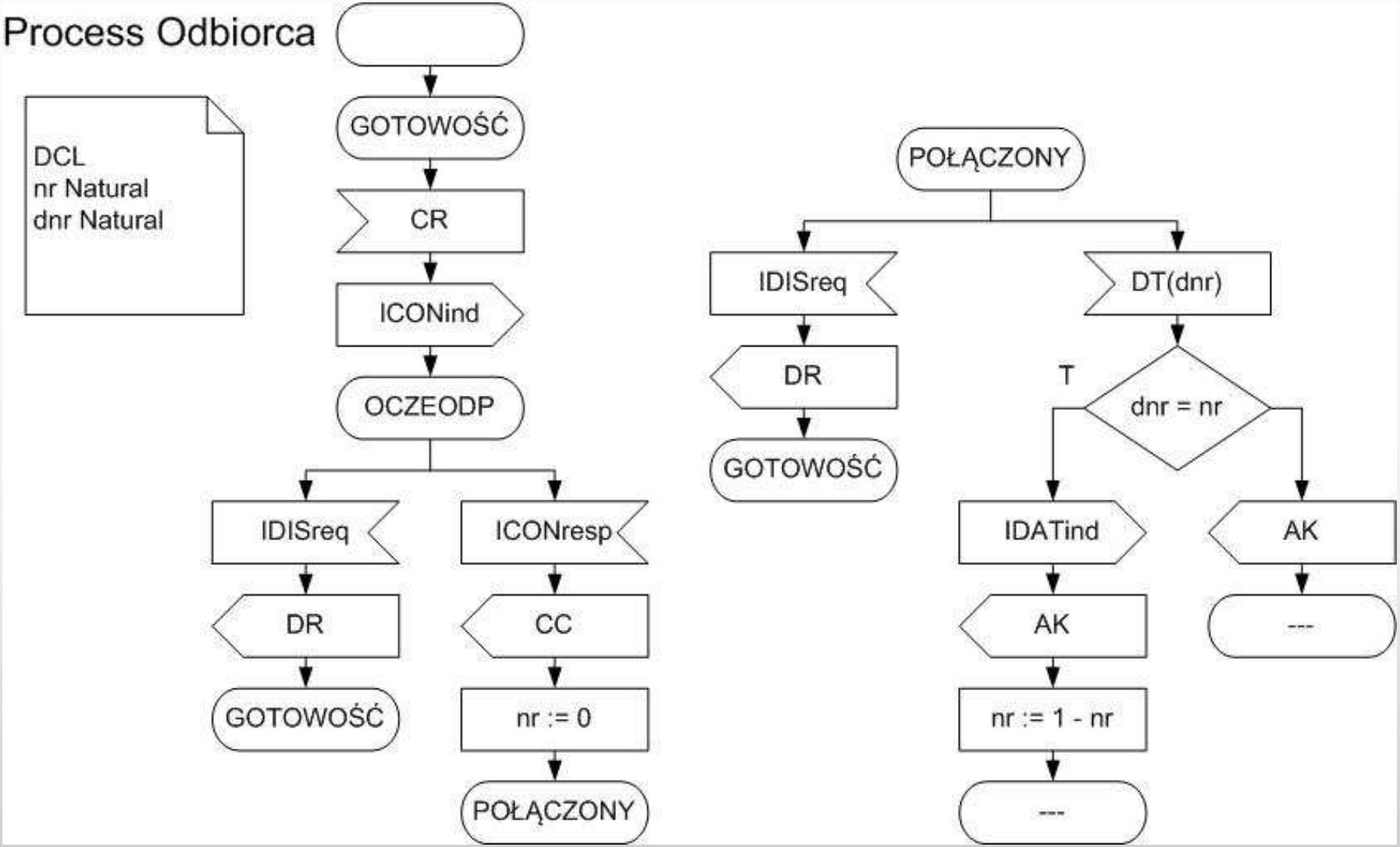
Protokół In-Res



SPECYFIKACJE FORMALNE

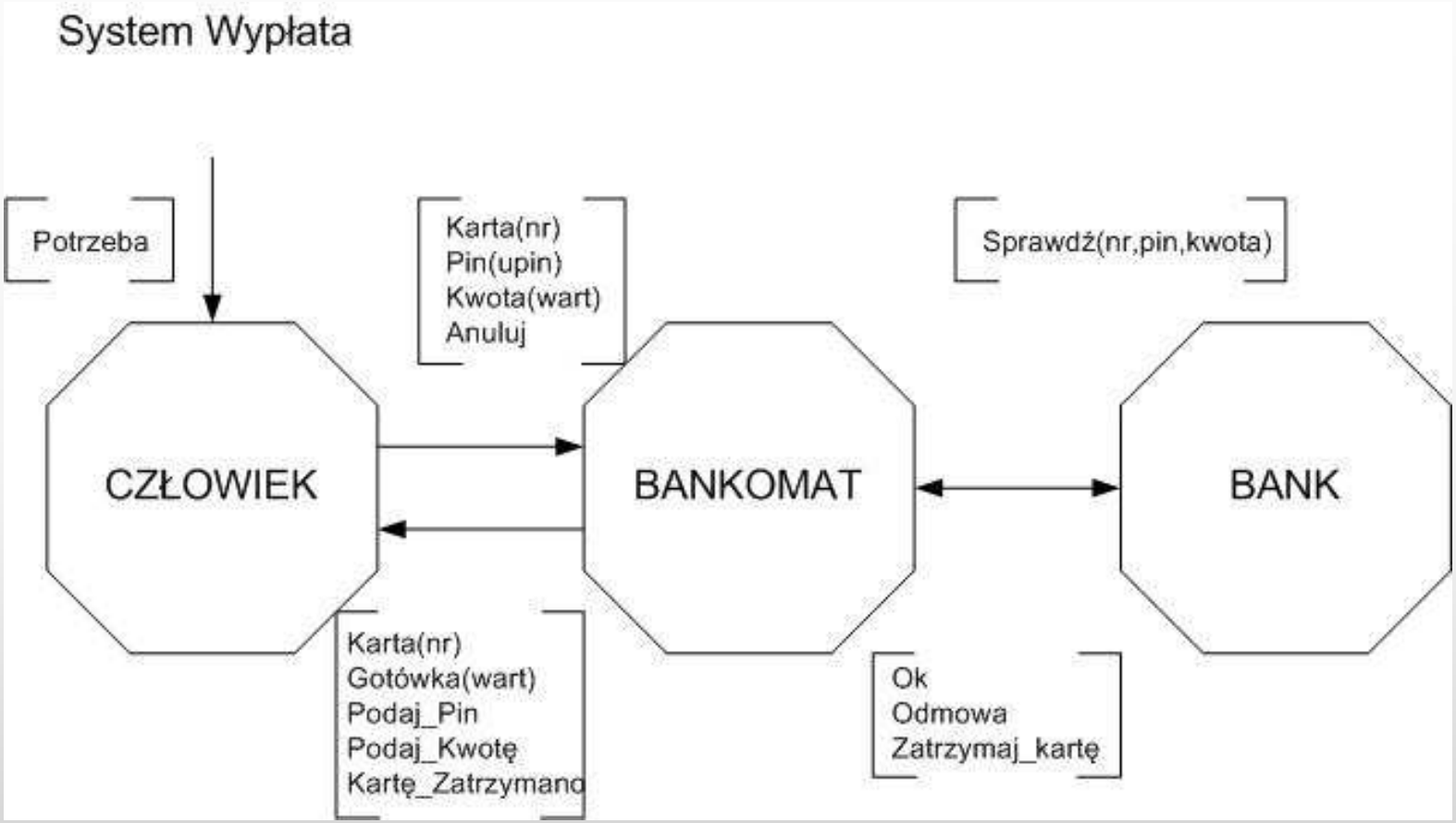


SPECYFIKACJE FORMALNE

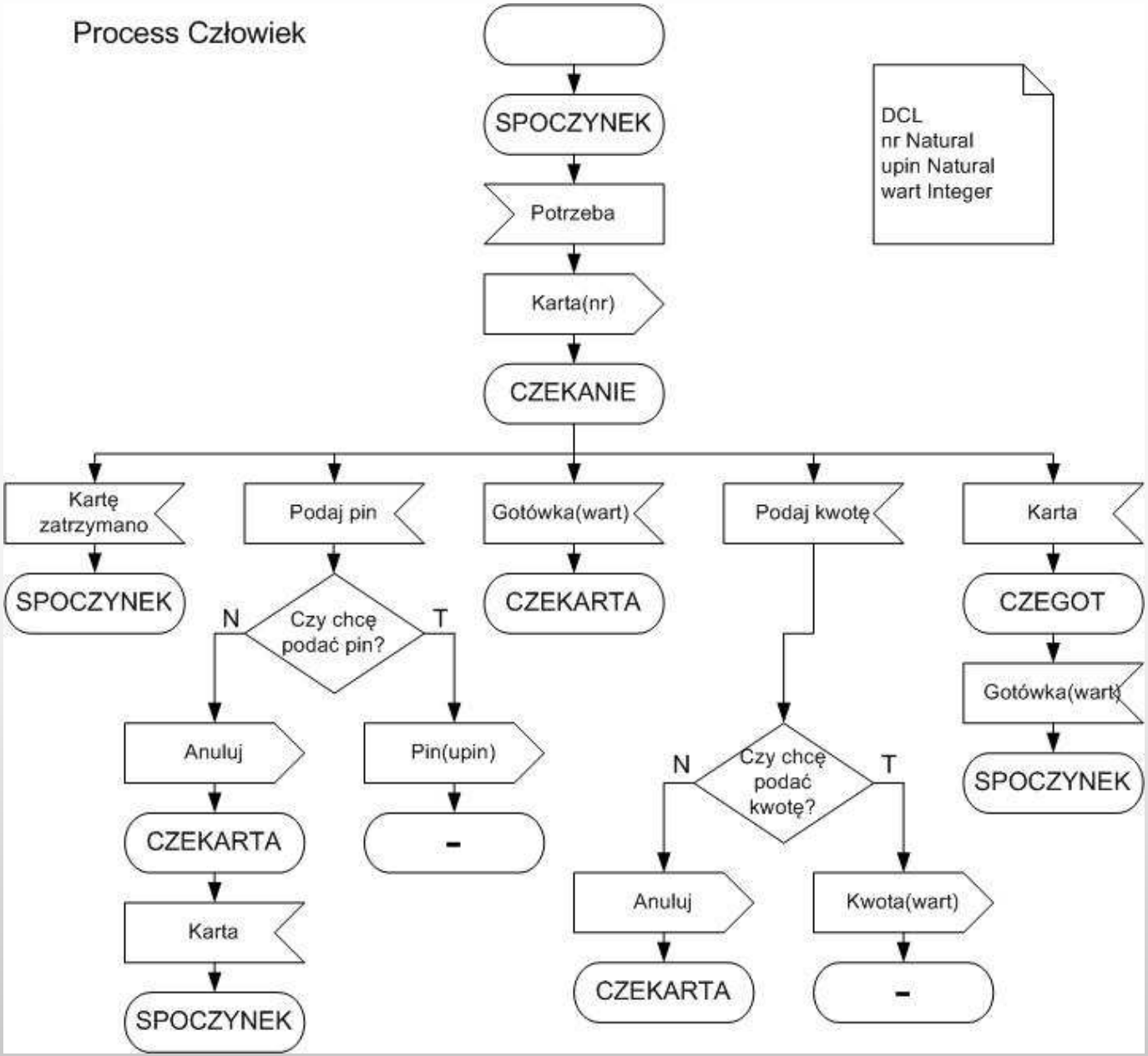


SPECYFIKACJE FORMALNE

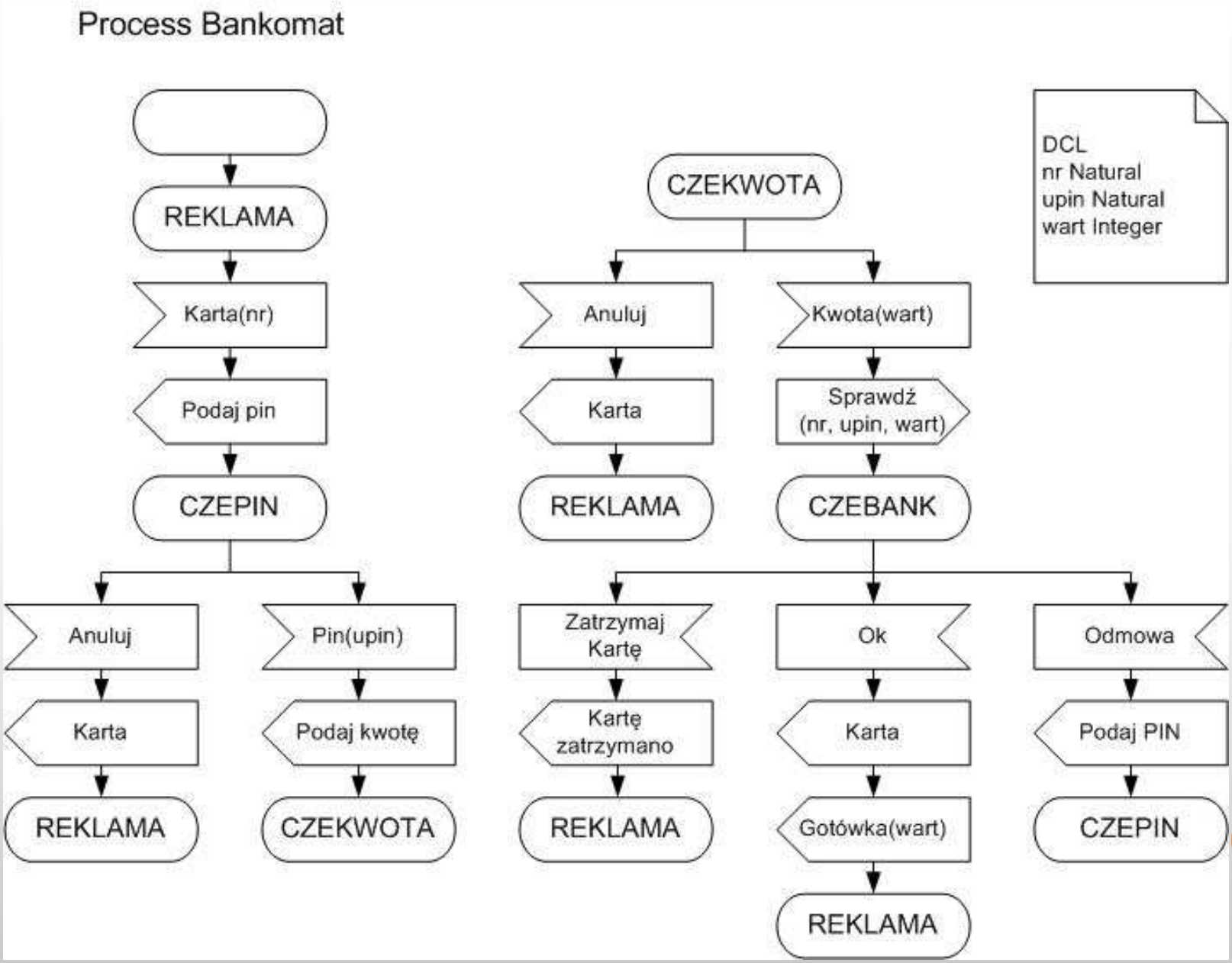
Protokół Wypłata (Klient-Bankomat-Bank)



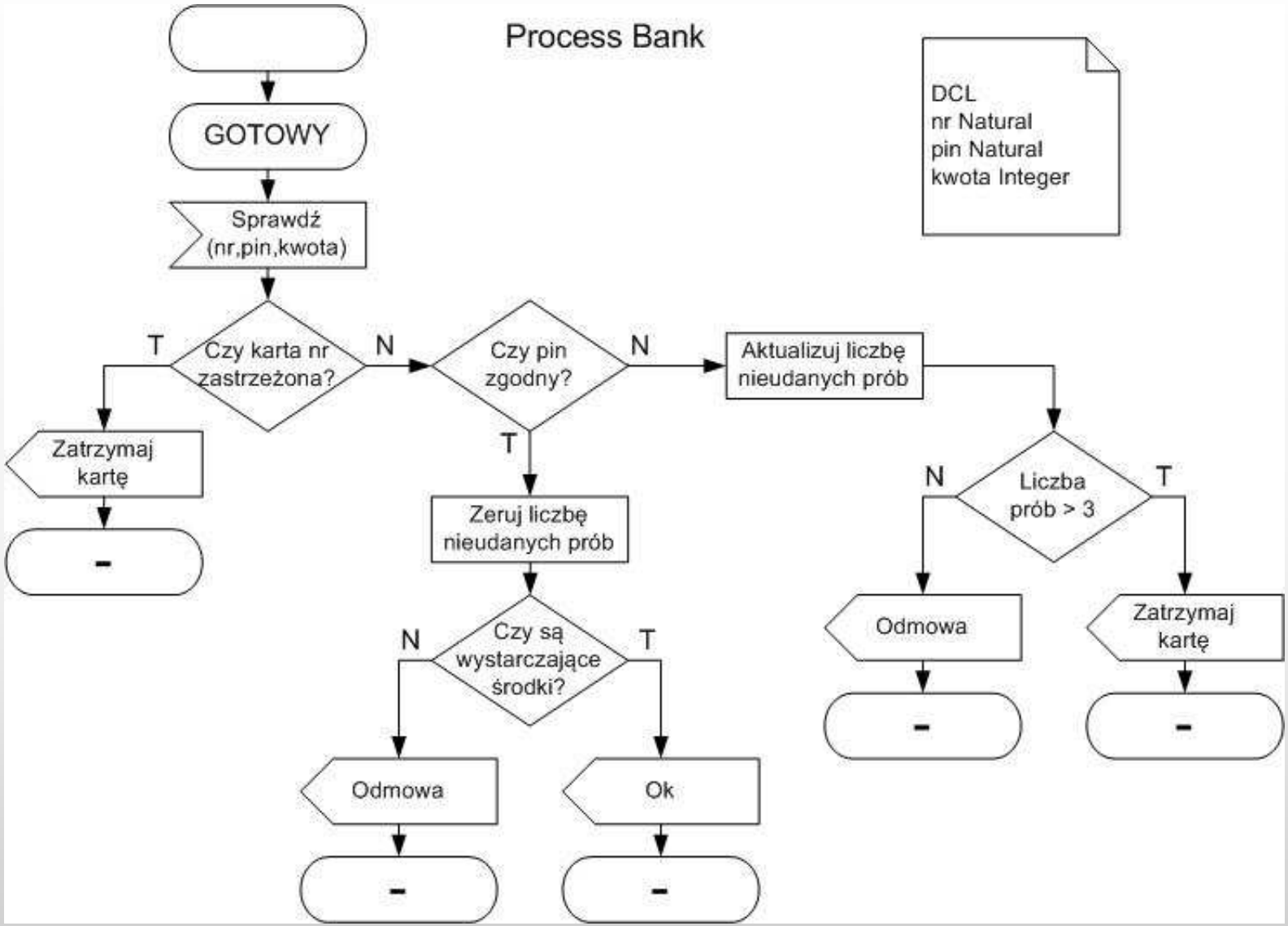
SPECYFIKACJE FORMALNE



SPECYFIKACJE FORMALNE



SPECYFIKACJE FORMALNE



SPECYFIKACJE FORMALNE

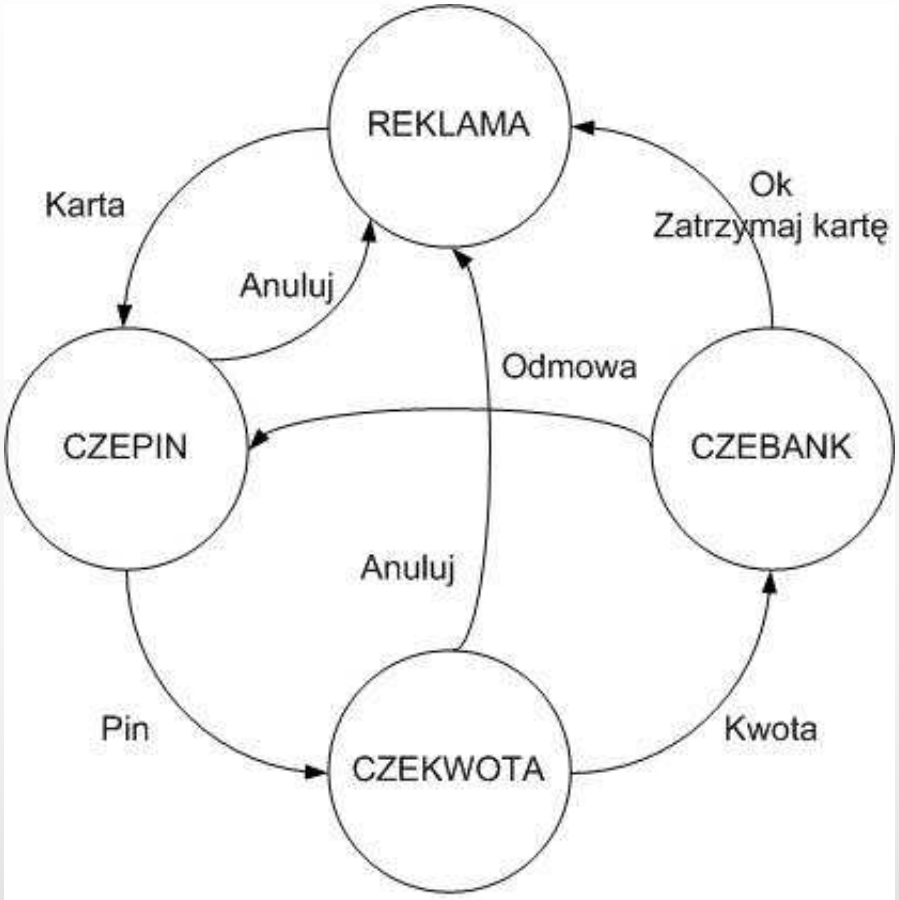
Diagram
stanów i przejść

oraz

Test zgodności

dla

Procesu Bankomat



Karta - Anuluj
- Karta - Pin - Anuluj
- Karta - Pin - Kwota - Odmowa
- Pin - Kwota - Ok
- Karta - Pin - Kwota - Zatrzymaj kartę