SIECI KOMPUTEROWE
wykład 15
Inżynieria protokołów

dr inż. Michał Sajkowski

literatura podstawowa
(w porządku chronologicznym)

G.V. Bochmann, C.A. Sunshine, „Formal Methods in Communication Protocol Design”, IEEE Trans. on Communications, vol 28, no 4, April 1980

P. Zafiropulo et al. „Towards Analyzing and Synthesizing Protocols”, IEEE Trans. on Communications, Vol 28, No 4, April 1980

D.P. Brand, P. Zafiropulo, „On Communicating Finite State Machines”, JACM, v 30, no 2, April 1983

T.F. Piatkowski, „Protocol Engineering”, ICC’83, 1983

M.G. Gouda, C.H. Chow, S.S. Lam, „Detection of Livelocks in Networks of Communicating Finite State Machines”, PSTV IV, IFIP 1985

M. Sajkowski, „Protocol Verification Techniques: Status Quo and Perspectives”, PSTV IV, IFIP 1985

literatura podstawowa
(w porządku chronologicznym)

M. Sajkowski, „On Verifying Time Dependent Protocols”, SETTS, 1986

J. Billington, „Protocol Engineering and Nets”, European PN Workshop 1987

H. Rudin, „Protocol Engineering: A Critical Assessment”, PSTV VIII, IFIP 1988

G.J. Holzmann, „Design and Validation of Computer Protocols, Prentice-Hall 1991

E.M. Clarke, R.P. Kurshan, „Computer-aided verification”, IEEE Spectrum, June 1996

G.J. Holzmann, „The Model Checker SPIN”, IEEE Trans on Software Engineering, vol 23, no 5, May 1997

M. Sajkowski, „Weryfikacja dynamicznych cech protokołów sieci komputerowych”, rozprawa doktorska, WE PP, Poznań 2000 (promotor Prof. Jerzy Brzeziński)

geneza inżynierii protokołów (1)

- od końca lat sześćdziesiątych trwa nieustanny rozwój sieci komputerowych, zarówno sprzętu jak i oprogramowania
- w zakresie oprogramowania, jednym z najistotniejszych kierunków jest rozwój szeroko rozumianych *algorytmów rozproszonych*, w tym również asynchronicznych systemów współbieżnych i protokołów sieci komputerowych
- protokoły sieci komputerowych charakteryzują się ogromną różnorodnością, wynikającą z wielości specyficznych wymagań, uwarunkowań historycznych, polityki wielkich firm informatycznych i stale pojawiających się nowych dziedzin zastosowań

geneza inżynierii protokołów (2)

- z teoretycznego punktu widzenia, protokoły to specyficzne systemy współbieżne, obejmujące zbiory współbieżnie wykonywanych procesów
- specyfika protokołu, jako systemu współbieżnego jest konsekwencją, z jednej strony, przyjętej *funkcjonalności* protokołu, a z drugiej, *własności środowiska*, w którym jest on wykonywany
- protokół definiuje i realizuje funkcje odpowiadające *regułom rządzącym wymianą komunikatów* przez kanały utworzone między dwoma lub więcej procesami, nazywanymi tu tradycyjnie *stacjami*

geneza inżynierii protokołów (3)

- protokół wykonywany jest w środowisku rozproszonym, w ogólności asynchronicznym, w którym brak jest współdzielonej pamięci i zegara, nie można ignorować niebezpieczeństwa awarii, a jedynym dostępnym mechanizmem komunikacji i synchronizacji jest wymiana komunikatów
- stąd też konstrukcja protokołów była i wciąż jest zadaniem bardzo złożonym, wymagającym silnego wsparcia ze strony *formalnych narzędzi matematycznych*, zarówno do opisu, jak i analizy zjawisk występujących w rzeczywistych protokołach
- doprowadziło to do powstania nowej dziedziny, nazwanej *inżynierią protokołów* - przez analogię do inżynierii oprogramowania - która podjęła to wyzwanie

definicja inżynierii protokołów

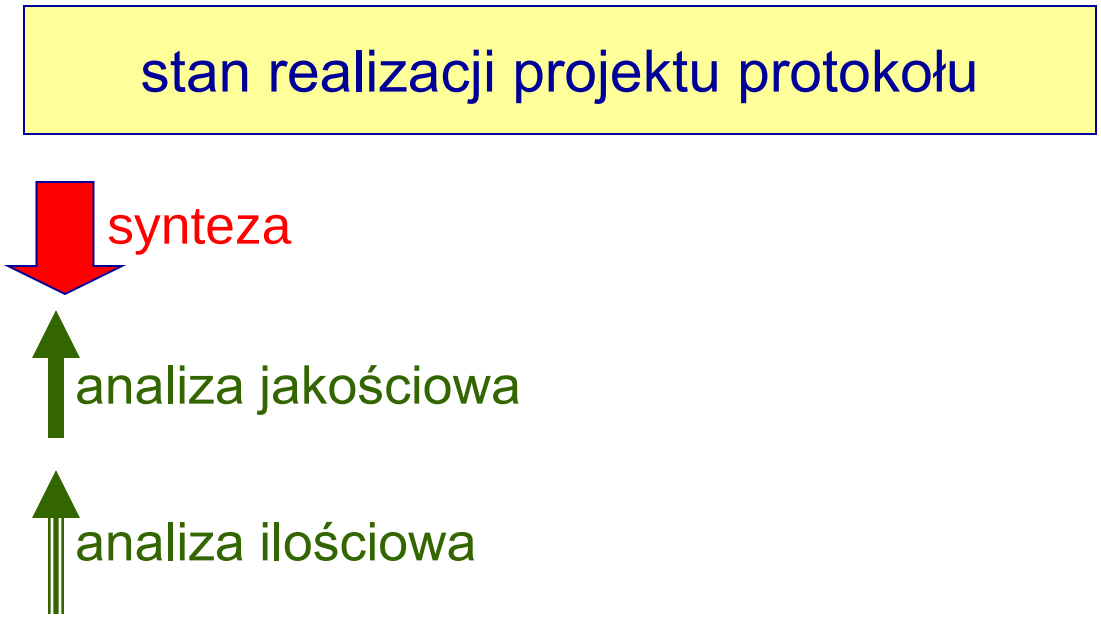
- inżynieria protokołów obejmuje całość działań związanych z wykorzystaniem metod formalnych i metod inżynierii oprogramowania w procesie tworzenia protokołu sieci komputerowej

działania podejmowane w inżynierii protokołów

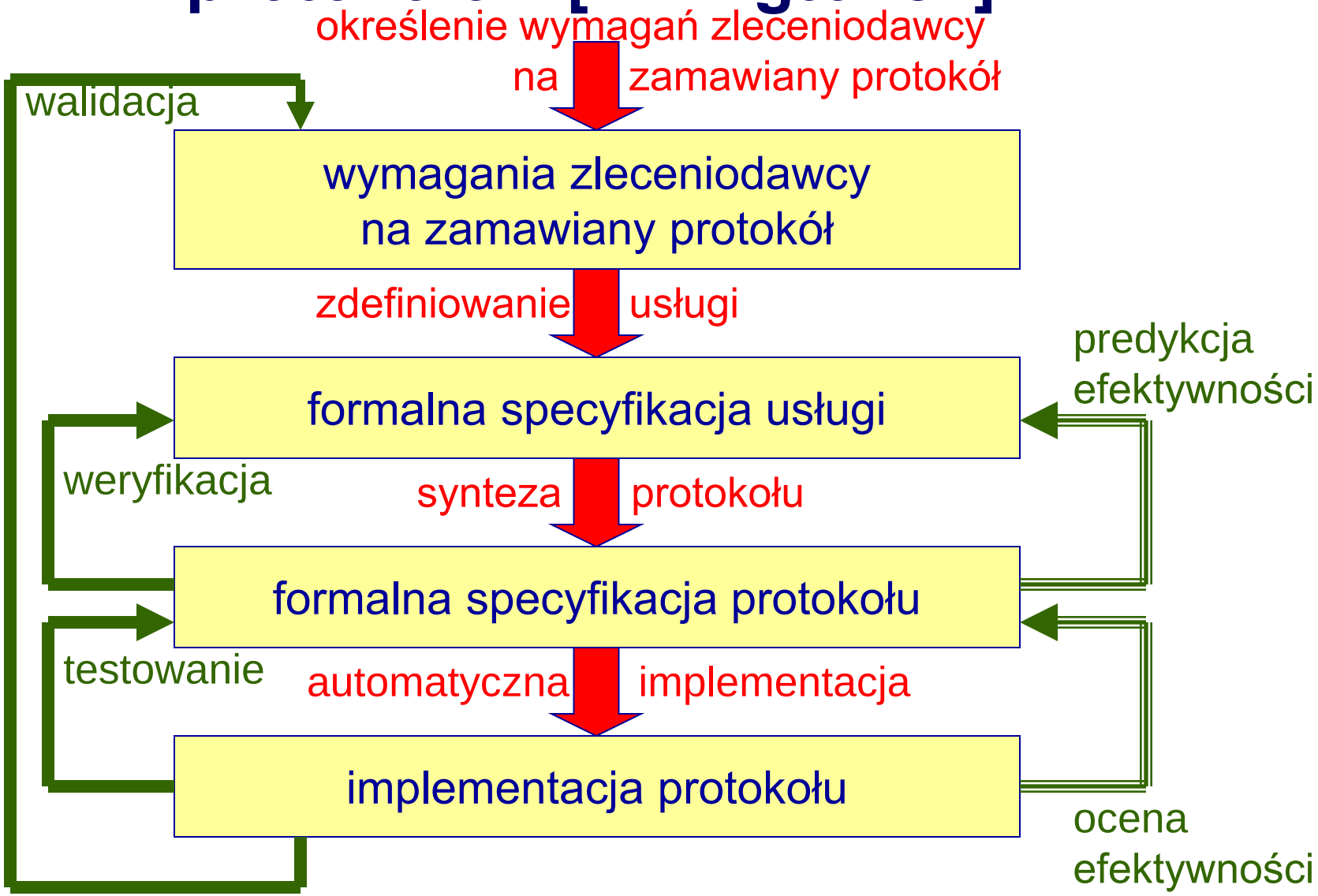
- w ogólności dzielą się na syntezę i analizę
- do syntezy należą:
  - określenie wymagań zleceniodawcy na zamawiany protokół (jest to punkt wyjściowy projektu protokołu)
  - formalne zdefiniowanie usługi
  - synteza protokołu
  - automatyczna implementacja protokołu
- analiza dzieli się na analizę jakościową i analizę ilościową
  - do analizy jakościowej należą: weryfikacja, testowanie i walidacja
  - do analizy ilościowej należą: predykcja efektywności i ocena efektywności

działania podejmowane w inżynierii protokołów

- **legenda**

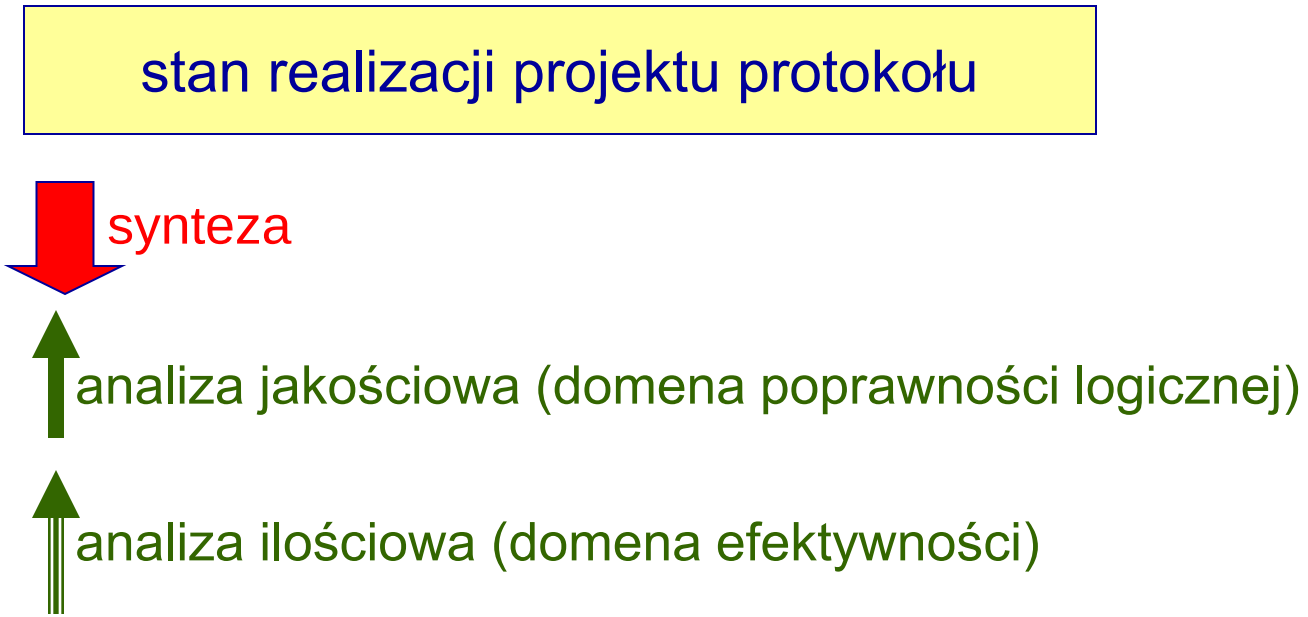


działania podejmowane w inżynierii protokołów [Billington87]

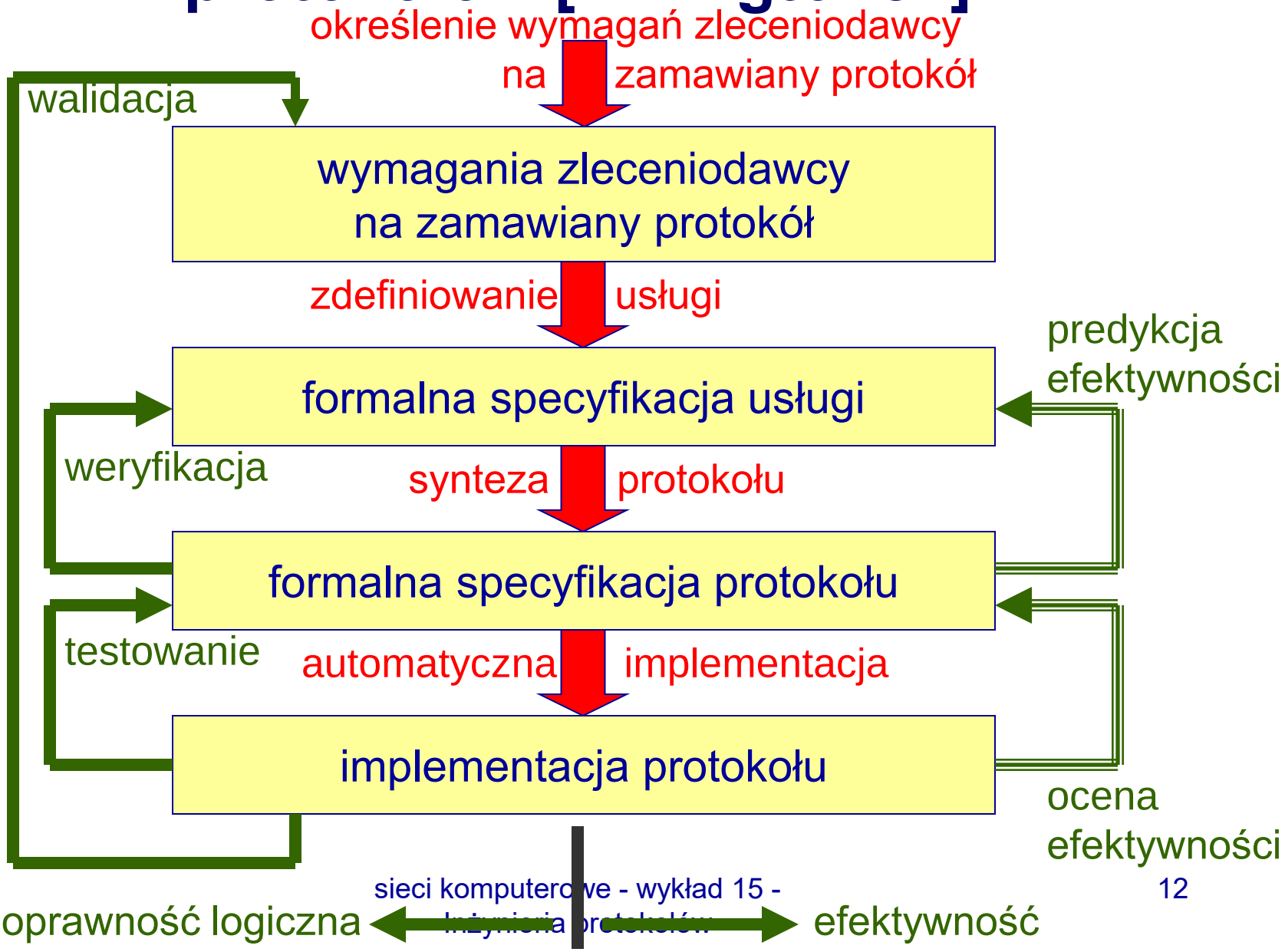


działania podejmowane w inżynierii protokołów

- **legenda**



działania podejmowane w inżynierii protokołów [Billington87]



działania podejmowane w inżynierii protokołów

- działania związane z syntezą:
- określenie wymagań zleceniodawcy na zamawiany protokół podaje w sposób nieformalny usługi, jakie protokół ma świadczyć i środowisko, w jakim ma pracować, może być wynikiem negocjacji między zamawiającym a projektantem protokołu
- zdefiniowanie usługi określa w sposób formalny interfejs usługobiorcy z protokołem
- synteza protokołu w sposób formalny wyprowadza specyfikację protokołu ze specyfikacji usługi
- automatyczna implementacja dokonuje transformację formalnej specyfikacji protokołu w implementację, za pomocą kompilatora

działania podejmowane w inżynierii protokołów

- działania związane z analizą:
- weryfikacja polega na zbadaniu poprawności formalnej specyfikacji protokołu i zgodności tej specyfikacji z formalną specyfikacją usługi
- testowanie bada zgodność implementacji z formalną specyfikacją protokołu
- walidacja bada zgodność implementacji z wymaganiami co do jakości, nałożonymi przez zleceniodawcę
- predykcja efektywności ocenia parametry ilościowe protokołu na podstawie jego formalnej specyfikacji
- ocena efektywności ocenia parametry na podstawie implementacji protokołu

ewolucja inżynierii protokołów

- z czasem inżynieria protokołów stała się jednym z głównych nurtów badawczych w sieciach komputerowych

ewolucja inżynierii protokołów (formalna specyfikacja)

- pierwsze prace w dziedzinie inżynierii protokołów dotyczyły modeli pozwalających w sposób matematyczny, czyli formalnie, opisać protokoły, ten kierunek badawczy był wyróżnikiem lat osiemdziesiątych
- pokazano, jak opisać protokoły za pomocą dziesiątek formalnych metod opisu (FDTs – formal description techniques), do metod tych należały: automaty, sieci Petriego, gramatyki, algebry procesów, logiki temporalne, języki specyfikacji, języki programowania i wiele innych
- w związku z pojawieniem się ogromnej liczby metod, zdecydowano się trzy z nich uznać za normę międzynarodową dla opisu protokołów komunikacyjnych: Estelle, Lotos i SDL

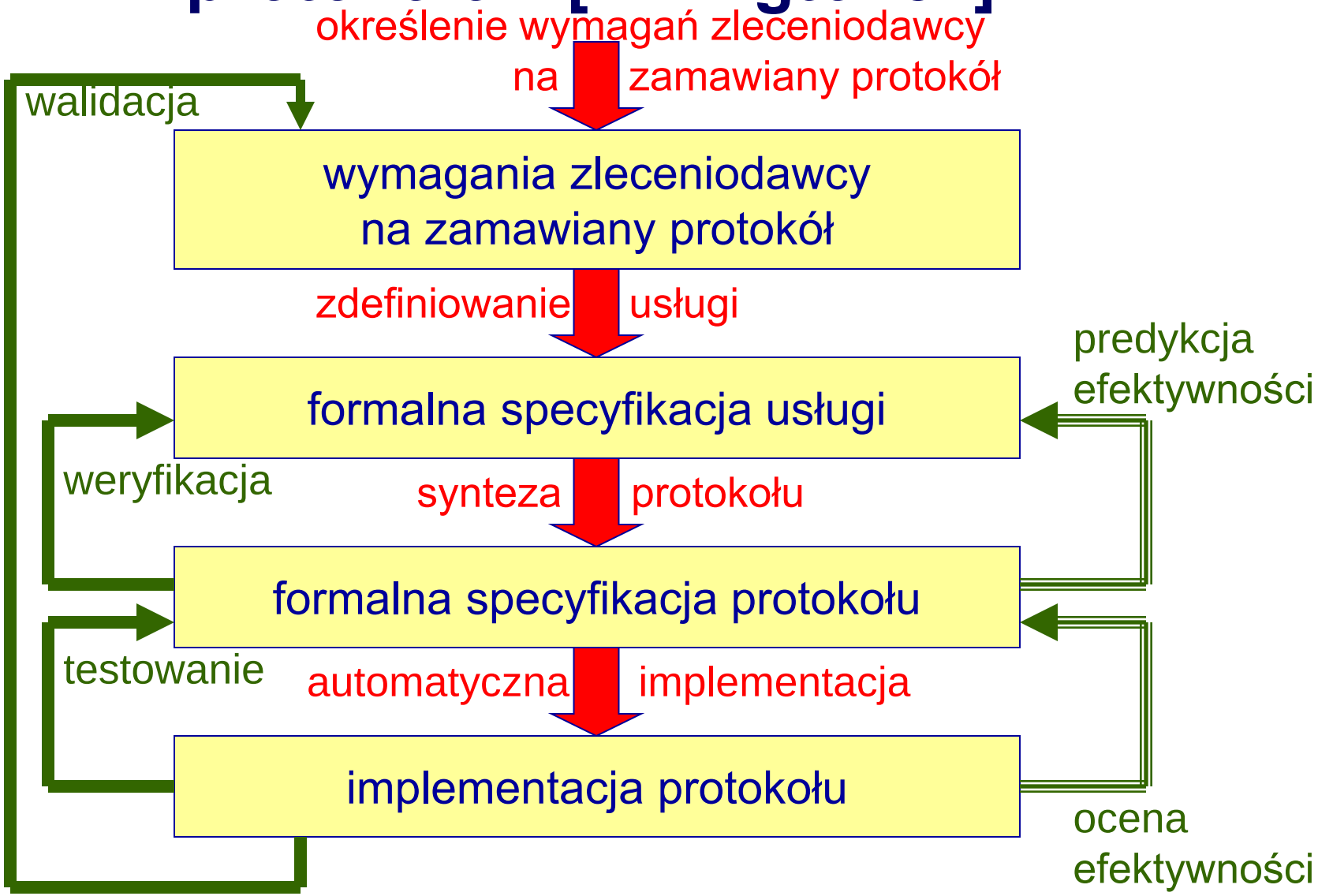
ewolucja inżynierii protokołów (testowanie)

- również na lata osiemdziesiąte, równolegle do prac nad specyfikacją, przypada gwałtowny rozwój metod testowania implementacji protokołów, zapoczątkowany pionierską konferencją zorganizowaną przez Dave'a Raynera z NPL w 1981
- wspomniany rozwój dotyczy również zagadnień architektury systemów testowania, jak samych metod testowania
- intensywne prace nad testowaniem kontynuowano w latach dziewięćdziesiątych, w Europie i w Stanach

ewolucja inżynierii protokołów (weryfikacja)

- szczególną rolę w inżynierii protokołów zyskała sobie weryfikacja, podejmowanie nierozwiązanych problemów w tej dziedzinie od samego początku miało wysoką rangę, zarówno w środowisku projektantów i badaczy protokołów, jak i użytkowników istniejących protokołów oraz, w szczególności, zleceniodawców i inwestorów nowych produktów – powód był i jest jeden – finanse
- weryfikacja jest pierwszą z przeprowadzanych analiz jakościowych w procesie projektowania protokołu, w związku z tym, nakłady na nią poniesione, zmniejszają ewentualne środki wydatkowane w późniejszych etapach, zwłaszcza na testowanie i walidację

działania podejmowane w inżynierii protokołów [Billington87]



ewolucja inżynierii protokołów (weryfikacja cech statycznych)

- weryfikacja była przedmiotem stałego zainteresowania badaczy od wielu lat
- w latach osiemdziesiątych metody weryfikacji protokołów skupiały się głównie na wykrywaniu niezależnych od czasu, statycznych cech protokołów, takich jak np. zakleszczenia
- pionierską pracę w tej dziedzinie wykonano w IBM Zurich, w tym również w zakresie pierwszych prób stosowania narzędzi automatycznej weryfikacji, przegląd i ocenę metod stosowanych w tamtych czasach zawiera praca [Sajkowski85]

ewolucja inżynierii protokołów (weryfikacja protokołów zależnych od czasu)

- znaczenie weryfikacji wzrosło jeszcze w latach dziewięćdziesiątych, w związku ze zwiększającym się zapotrzebowaniem na protokoły, spełniające określone wymagania czasowe i efektywnościowe, zwłaszcza w systemach czasu rzeczywistego i w hybrydowych (dyskretno-ciągłych) systemach sterowania
- nowe zastosowania, odnoszące się do zależności czasowych, wymagają bardziej złożonych i dokładnych modeli, prowadzą więc do większych przestrzeni stanów poddanych analizie

ewolucja inżynierii protokołów (komputerowo wspomagana weryfikacja)

- doprowadziło to, w konsekwencji, do bardzo szybkiego rozwoju metod formalnych, narzędzi programowych i algorytmów, stosowanych do komputerowo wspomaganej weryfikacji systemów współbieżnych czasu rzeczywistego, a w szczególności protokołów sieci komputerowych
- rozwój tych metod był tak intensywny, że komputerowo wspomagana weryfikacja protokołów stała się wyróżnikiem lat dziewięćdziesiątych w swojej dziedzinie
- osiągnięcia lat dziewięćdziesiątych to głównie prace nad narzędziami automatycznej weryfikacji protokołów, np. nad profesjonalnym systemem SPIN [Holzmann91]

ewolucja inżynierii protokołów (weryfikacja cech dynamicznych)

- na przełomie lat osiemdziesiątych i dziewięćdziesiątych podejmowano próby wykrywania cech dynamicznych w protokołach, stosowanych w sieciach komputerowych, w systemach czasu rzeczywistego i dyskretno-ciągłych systemach sterowania
- cechy dynamiczne dotyczą własności, których charakter, albo wręcz obecność, zależy od relacji czasowych występujących między zdarzeniami w protokole
- w celu weryfikacji cech dynamicznych, wprowadzono do modeli formalnych protokołów czas w postaci jawnej, zasadniczą zaletą takich modeli, w porównaniu do modeli bez czasu, jest możliwość dokładniejszego opisywania zjawisk występujących w protokołach i w rezultacie wierniejsze odtworzenie przestrzeni stanów osiąganych w rzeczywistym protokole

pojęcia podstawowe - protokół

- nieformalnie, protokołem komunikacyjnym, lub krótko protokołem, nazywamy reguły rządzące wymianą komunikatów przez kanały utworzone między dwoma (lub więcej niż dwoma procesami)
- taki proces, zaangażowany w protokół nazywamy stacją
- formalnie, protokołem nazywamy zbiór stacji:

$$Pr = <ent_1, ent_2, ..., ent_3>$$

- zauważmy, że zbiór kanałów utworzony między dwoma stacjami nie pojawia się jawnie w definicji protokołu - jest tak dlatego, ponieważ specyfikacja każdej ze stacji już określa, które ze stacji są połączone za pomocą poszczególnych kanałów, oraz które komunikaty są przesyłane w określonym kierunku

pojęcia podstawowe – automat skończony

- stację protokołu można zdefiniować formalnie za pomocą pojęcia *komunikującego się automatu skończonego*
- rozpoczniemy jednak od pojęcia *automatu skończonego*, którym nazywamy czwórkę:

$$X = (S, E, \delta, q_0)$$

w której:

- S jest *zbiorem stanów*
- E jest *zbiorem zdarzeń*, które powodują zmianę stanu
- δ jest funkcją częściową, $\delta : S \times E \rightarrow S$, zwaną *funkcją przejścia*, określa ona następny stan automatu, oznaczony przez $\delta(s, e)$, do którego automat przechodzi ze stanu s pod wpływem zdarzenia e
- q_0 jest *stanem początkowym*, $q_0 \in S$

pojęcia podstawowe – komunikujący się automat skończony

- uwzględniając fakt, że automat skończony może porozumiewać się z innym automatem skończonym, za pomocą wymiany komunikatów, w naturalny sposób dochodzi się do pojęcia *komunikującego się automatu skończonego* (CFSM), którym nazywamy automat skończony połączony z innym automatem skończonym za pomocą kolejek wejściowych i wyjściowych, taki, w którym zbiór zdarzeń *E* dzieli się na *zbiór zdarzeń nadania komunikatu –E*, *zbiór zdarzeń odbioru komunikatu +E* i *zbiór zdarzeń wewnętrznych iE*

pojęcia podstawowe – stacja protokołu

- pojęcie CFSM jest podstawą pojęcia stacji protokołu
- *stacją protokołu* nazywamy komunikujący się automat skończony:

$$ent_I = \langle ES_I, -E_I, +E_I, iE_I, \delta_I, es_{0,I} \rangle$$

- w którym:
- ES_I jest *zbiorem stanów stacji*
- $-E_I$ jest *zbiorem zdarzeń nadania komunikatu*, komunikat nadawany przez stację ent_I do stacji ent_J oznaczamy przez x_{IJ}, a zdarzenie nadania komunikatu x_{IJ} przez $-x_{IJ}$

pojęcia podstawowe – stacja protokołu

- $+E_I$ jest *zbiorem zdarzeń odbioru komunikatu*, komunikat odbierany przez stację ent_I od stacji ent_J oznaczamy przez x_{JI}, a zdarzenie odbioru komunikatu x_{JI} przez $+x_{JI}$
- iE_I jest *zbiorem zdarzeń wewnętrznych stacji* ent_I, zdarzenie wewnętrzne stacji ent_I oznaczamy przez x_I
- δ_I jest *funkcją przejścia*, $\delta_I : ES_I \times (-E_I \cup +E_I \cup iE_I)$, która definiuje stan następny, do którego stacja przechodzi ze stanu bieżącego, po wystąpieniu zdarzenia nadania komunikatu, zdarzenia odbioru komunikatu, lub zdarzenia wewnętrznego
- $es_{0,I}$ jest *stanem początkowym stacji*, $es_{0,I} \in ES_I$

pojęcia podstawowe – stan protokołu

- aby można było coś powiedzieć na temat *zachowania się* protokołu, należy znać *stan protokołu*
- na stan protokołu można patrzeć jak na *stan wszystkich jego stacji i kanałów*
- *stanem* s_k protokołu nazywamy parę:

$$\langle S, C \rangle = \langle (es_1, es_2, ..., es_K), (cs_{12}, cs_{21}, ..., cs_{IJ}, cs_{JI}, ..., cs_{K,K-1}, ..., cs_{K-1,K}) \rangle$$

- gdzie es_I jest *stanem stacji* ent_I, a cs_{IJ} jest *stanem kanału* od stacji ent_I do stacji ent_J
- zazwyczaj, stan protokołu nazywa się *stanem globalnym*, a stany stacji i kanałów nazywa się *stanami lokalnymi*

pojęcia podstawowe – funkcja następnego stanu

- projektant protokołu powinien wiedzieć, do jakich kolejnych stanów może przejść protokół, może to osiągnąć korzystając z *funkcji następnego stanu*:
- jest to *funkcja częściowa* Δ taka, że $\forall I$ i $\forall J$, $I \neq J$, i $\forall e_i \in E_I \cup E_J$:

i. jeżeli $e_i = -x_{IJ}$ to $(S,C)' = \Delta((S,C),-x_{IJ})$ wtedy i tylko wtedy, gdy $es_i' = \delta_I(es_I,-x_{IJ})$ i $c_{IJ}' = c_{IJ} \bullet x_{IJ}$

ii. jeżeli $e_i = +x_{IJ}$ to $(S,C)' = \Delta((S,C),+x_{IJ})$ wtedy i tylko wtedy, gdy $es_j' = \delta_j(es_j,+x_{IJ})$ i $c_{IJ} = x_{IJ} \bullet c_{IJ}'$

iii. jeżeli $e_i = x_{I(J)}$ to $(S,C)' = \Delta((S,C),x_{I(J)})$ wtedy i tylko wtedy, gdy $es_{I(j)}' = \delta_{I(J)}(es_{I(j)},x_{I(J)})$ i $c_{IJ}' = c_{IJ}$ i $c_{JI}' = c_{JI}$

znak • oznacza *operację konkatenacji*

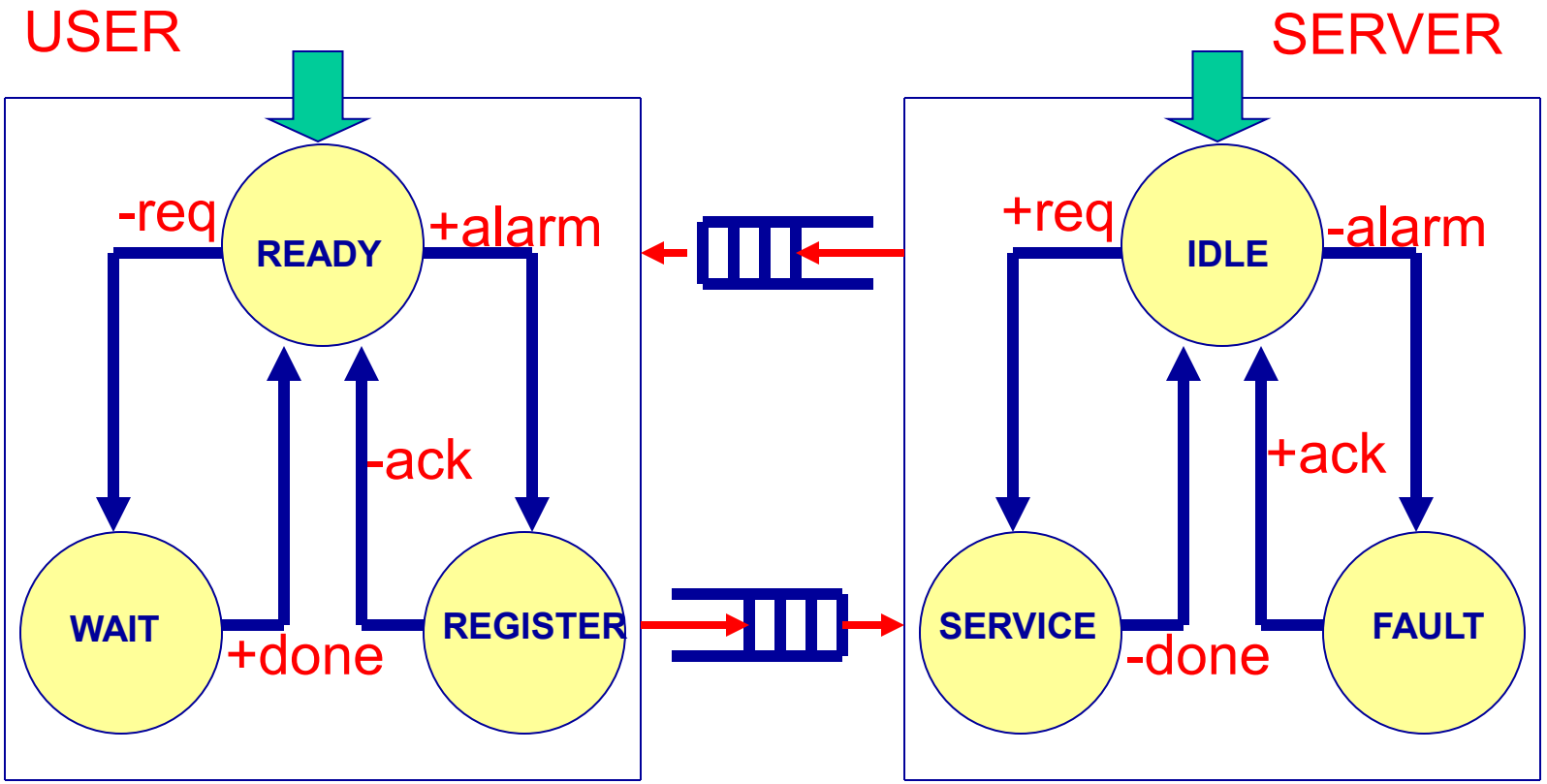
pojęcia podstawowe – relacja osiągalności

- funkcję następnego stanu można łatwo rozszerzyć do *relacji osiągalności Δ^* między stanami*, którą nazywamy zwrotne i przechodnie domknięcie funkcji następnego stanu
- każdy stan s_k *znajdujący się w relacji osiągalności* ze stanem początkowym s_0 jest z niego osiągalny. Stan taki nazywamy osiągalnym:
- *stanem osiągalnym* nazywamy stan s_k, taki, że $s_0 \ \Delta^* s_k$, gdzie s_0 jest stanem początkowym

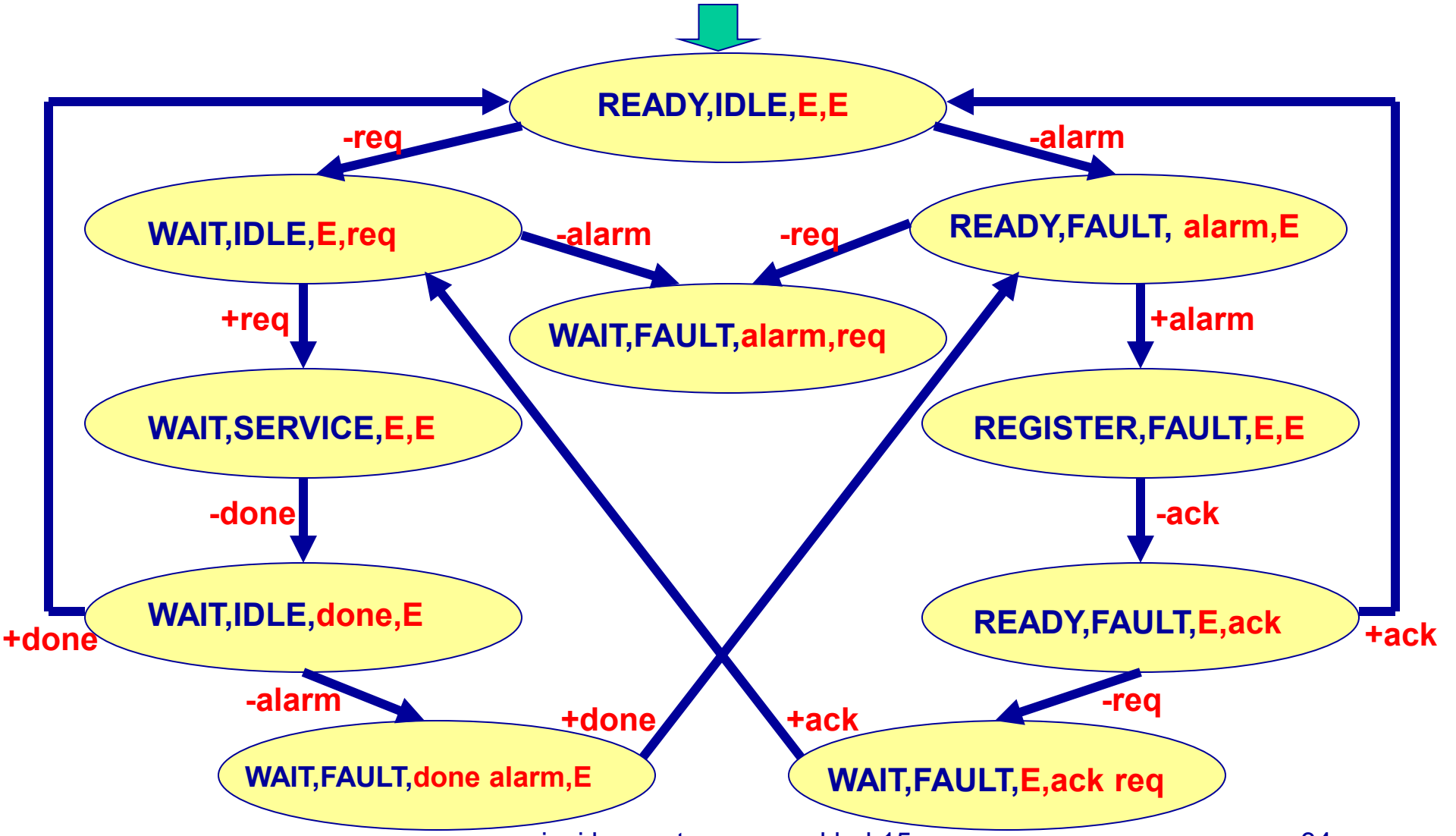
pojęcia podstawowe – przestrzeń stanów

- zbiór wszystkich stanów *osiągalnych* przez protokół nazywamy *przestrzenią stanów*
- przestrzeń ta występuje, między innymi, w postaci *grafu stanów osiągalnych*, zwanych zwyczajowo *grafem osiągalności*, jest to graf skierowany, którego wierzchołki odpowiadają *stanom protokołu*, a krawędzie *przejściom między stanami*
- po to, aby w procesie generacji przestrzeni stanów uzyskać graf, należy zidentyfikować *stany równoważne*, które można następnie zagregować do do pojedynczego wierzchołka w grafie
- dwa stany uznajemy za równoważne jeżeli ich reprezentacja jest *identyczna*

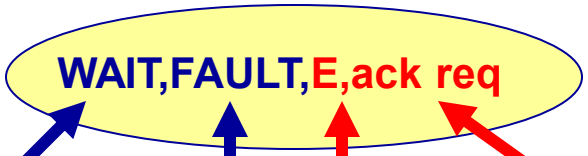
przykład opisu protokołu – User Server Protocol (USP)



graf stanów osiągalnych protokołu USP



stan globalny w protokole USP - legenda



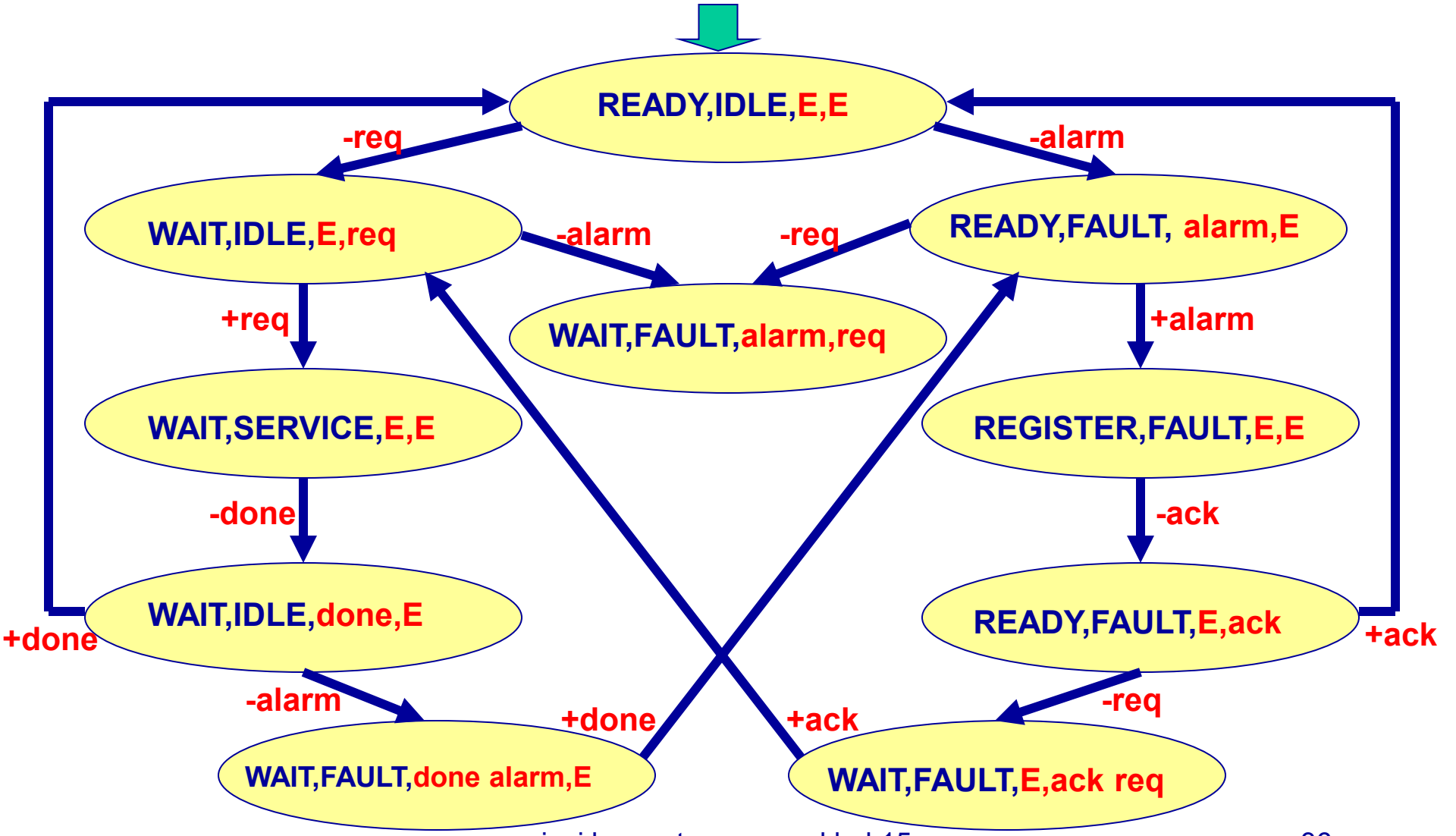
stan stacji User
(tutaj WAIT)

stan stacji Server
(tutaj FAULT)

stan kolejki (kanału)
wejściowej stacji User
(tutaj E – kolejka pusta)

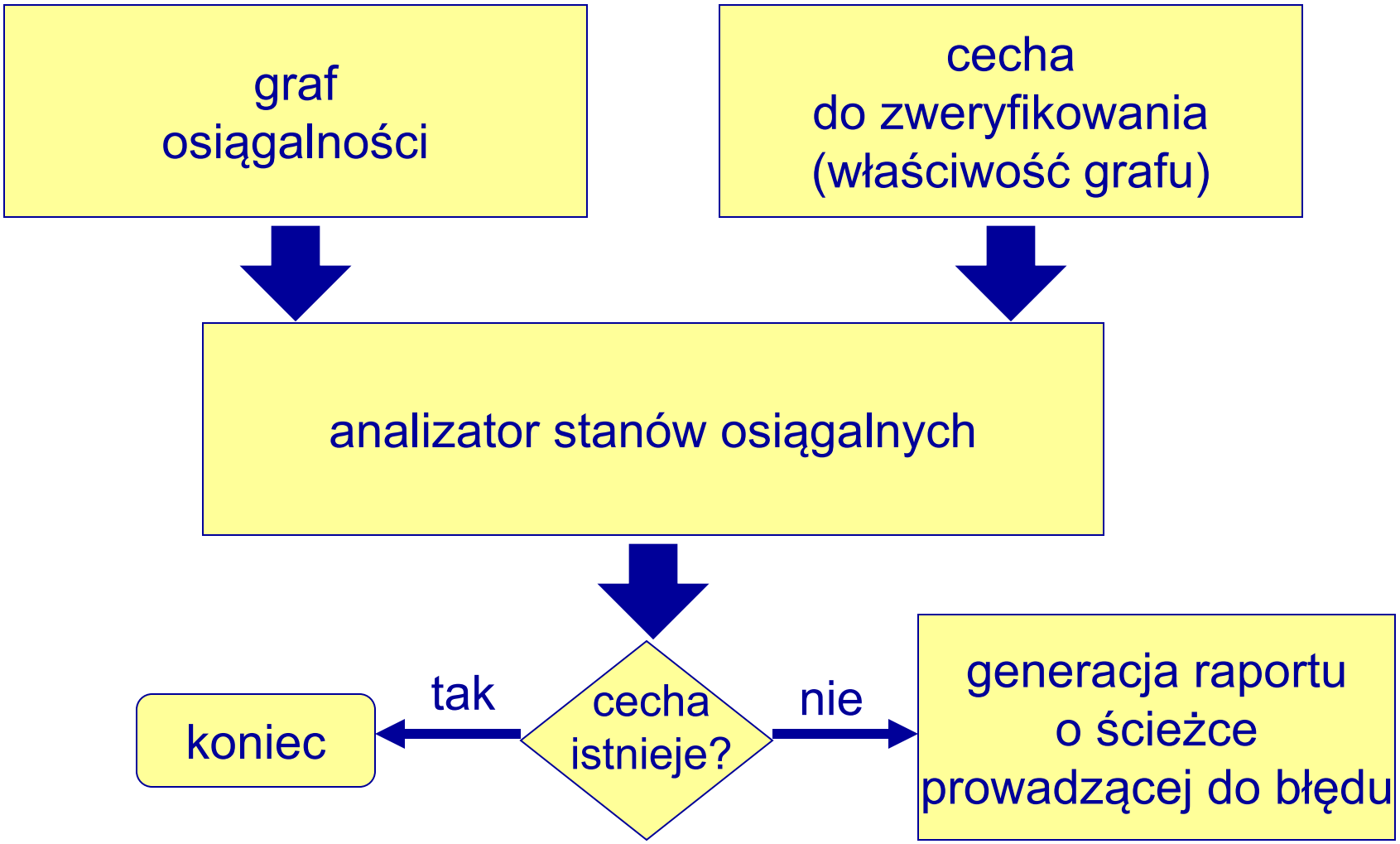
stan kolejki (kanału)
wejściowej stacji Server
(tutaj dwa komunikaty
w kolejce wejściowej:
ack oraz req)

graf stanów osiągalnych protokołu USP



weryfikacja

analiza osiągalności



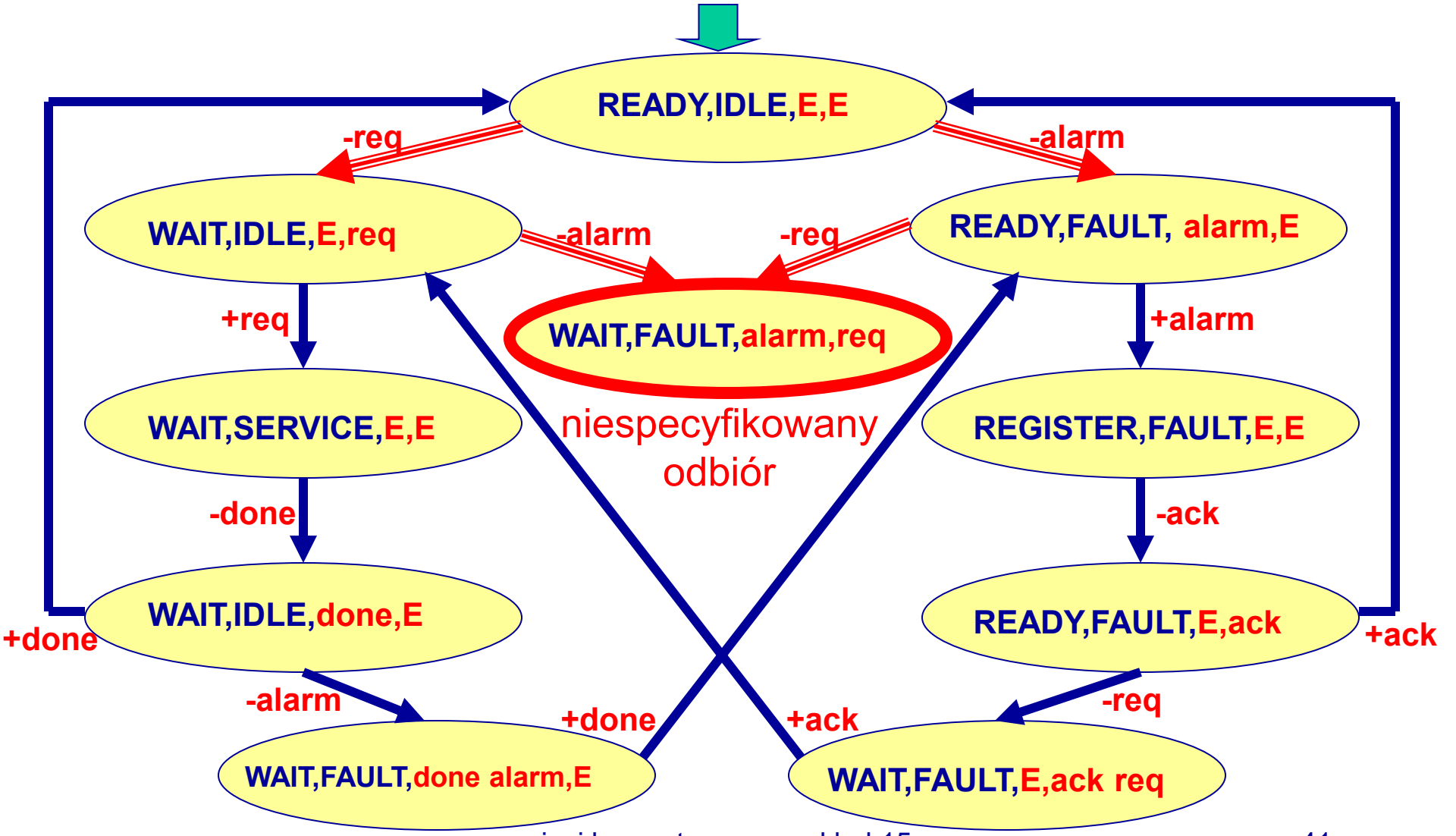
cechy statyczne protokołu

- zakleszczenie – stan, w którym nie są możliwe żadne zdarzenia nadawania, a kanały są puste
- niespecyfikowany odbiór – odbiór, który występuje w rzeczywistości nie jest umieszczony w projekcie
- niewykonalna interakcja – projekt zawiera zdarzenia nadawania i odbioru, które nie mogą wystąpić w rzeczywistości
- dwuznaczność stanów – stan lokalny jednej stacji współistnieje stabilnie z kilkoma innymi stanami lokalnymi w drugiej stacji – stan stabilny, to taki, w którym oba kanały są puste

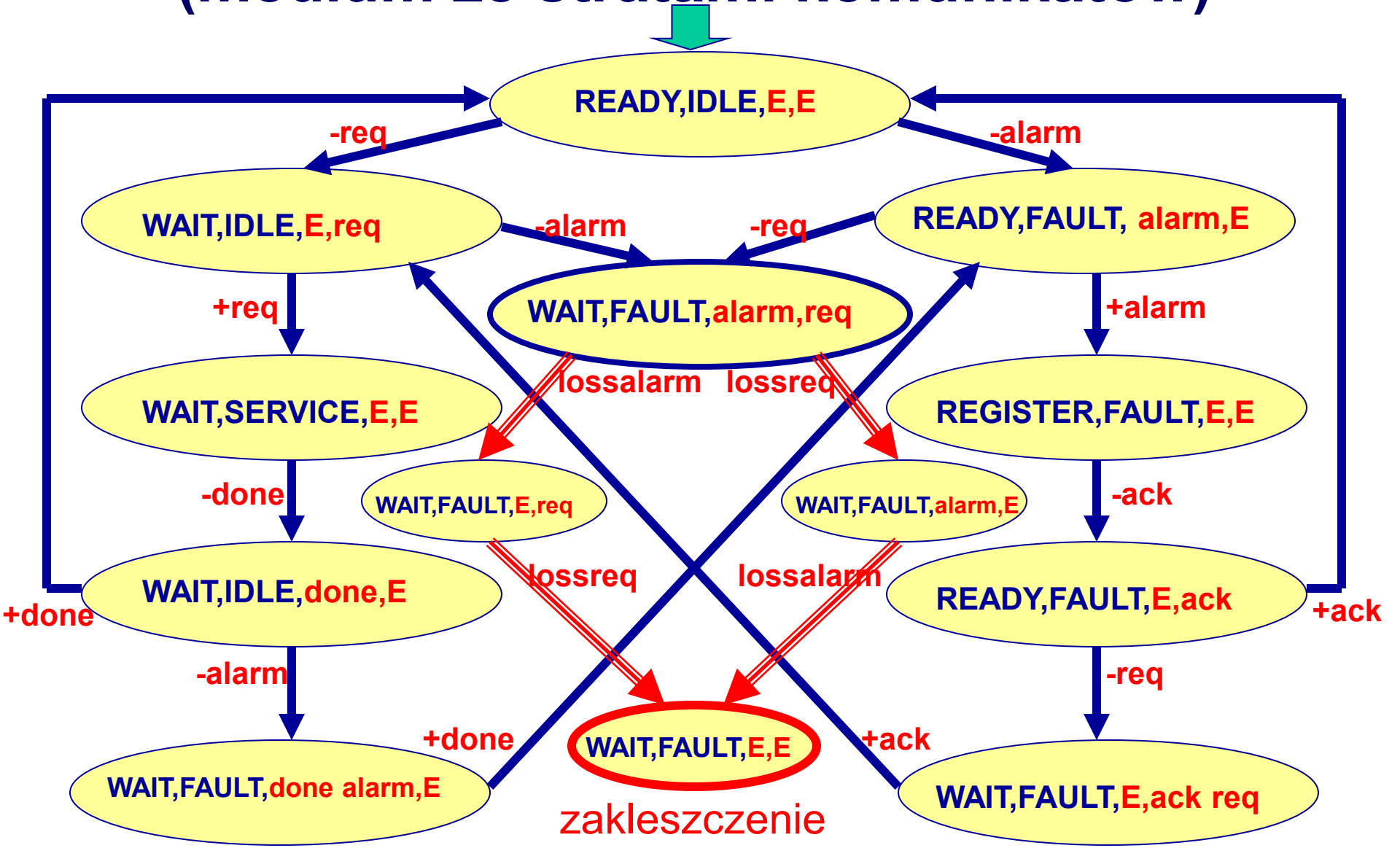
przykłady cech statycznych protokołu

- dwie z tych cech pokażemy na przykładzie protokołu USP: niespecyfikowany odbiór oraz zakleszczenie

niespecyfikowany odbiór w protokole USP



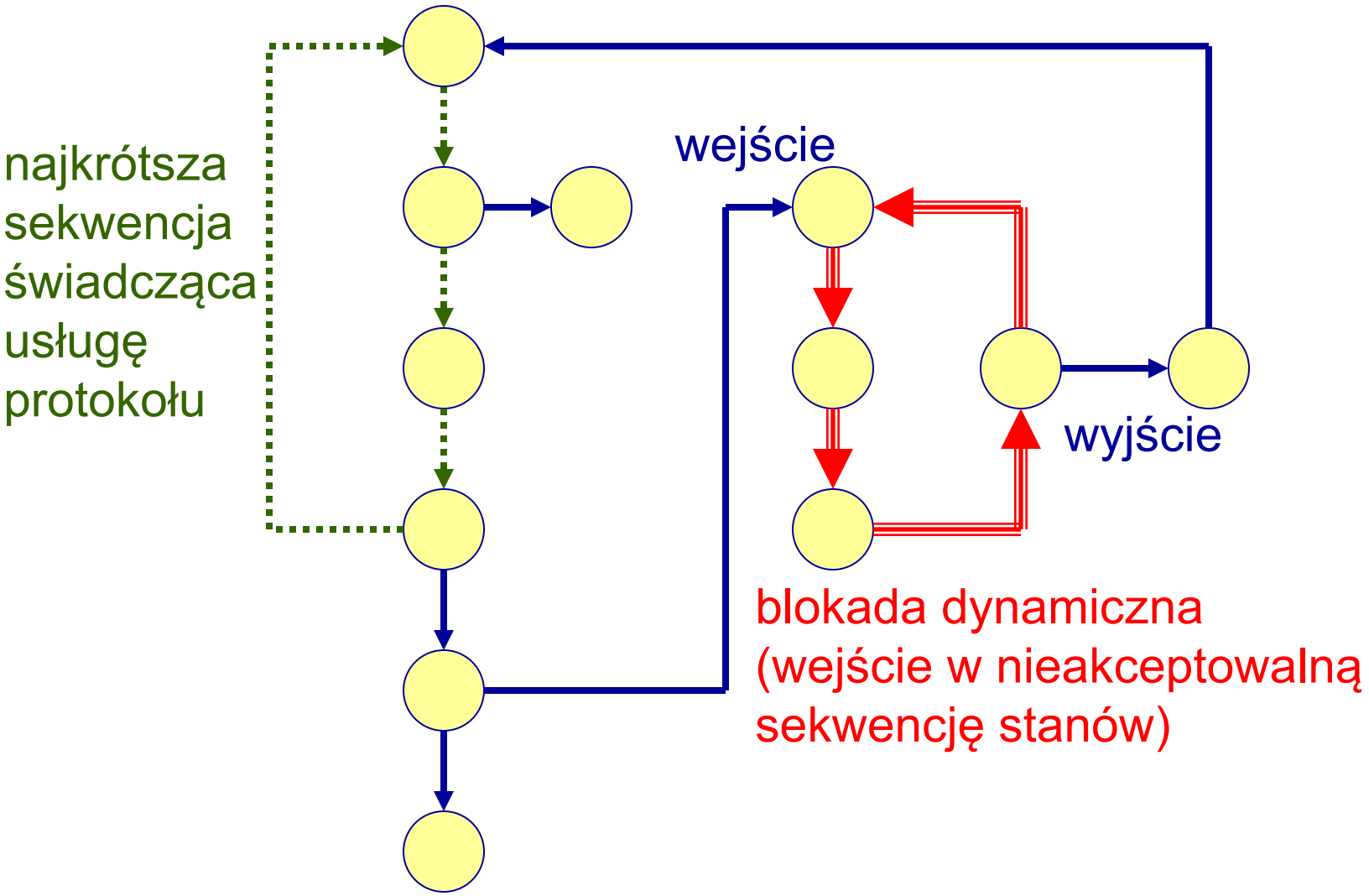
zakleszczenie w protokole USP (medium ze stratami komunikatów)



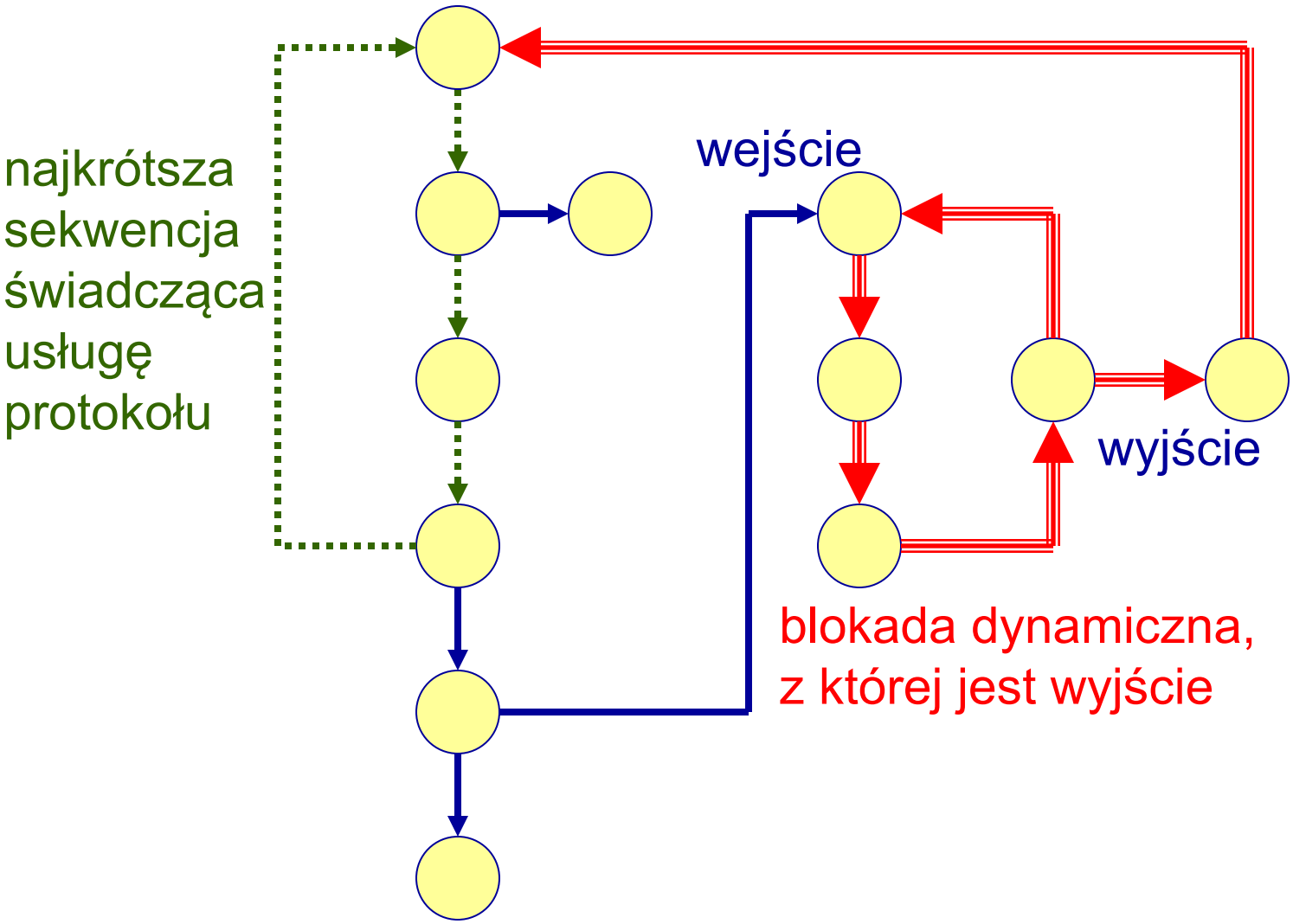
cechy dynamiczne protokołu

- blokada dynamiczna – wejście w nieakceptowalną sekwencję stanów
- zakleszczenie dynamiczne – blokada dynamiczna, z której nie ma wyjścia
- blokowanie tempa – blokada dynamiczna, z której jest wyjście

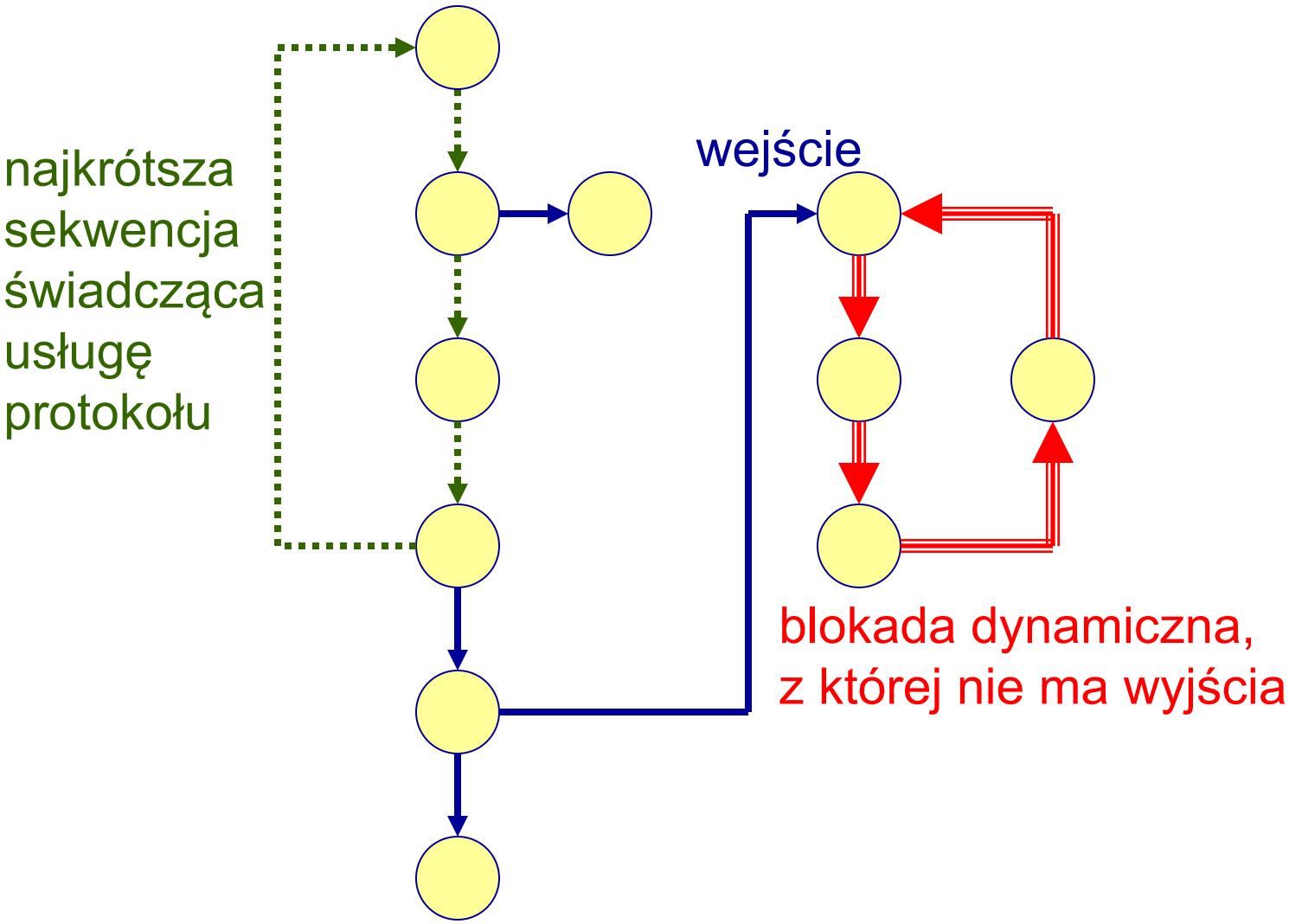
blokada dynamiczna



blokowanie tempa

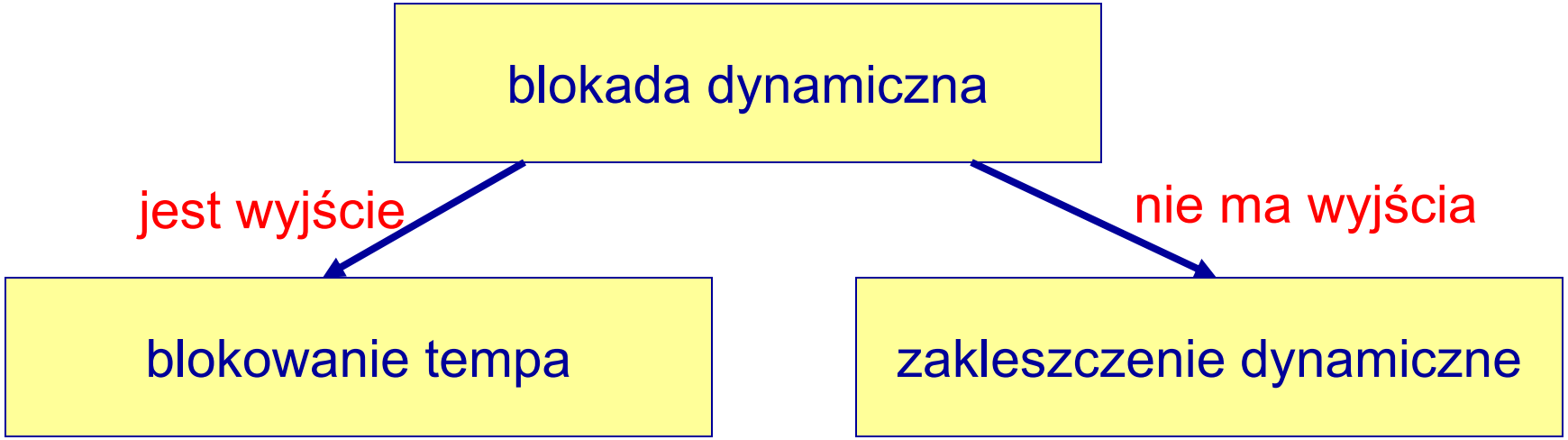


zakleszczenie dynamiczne

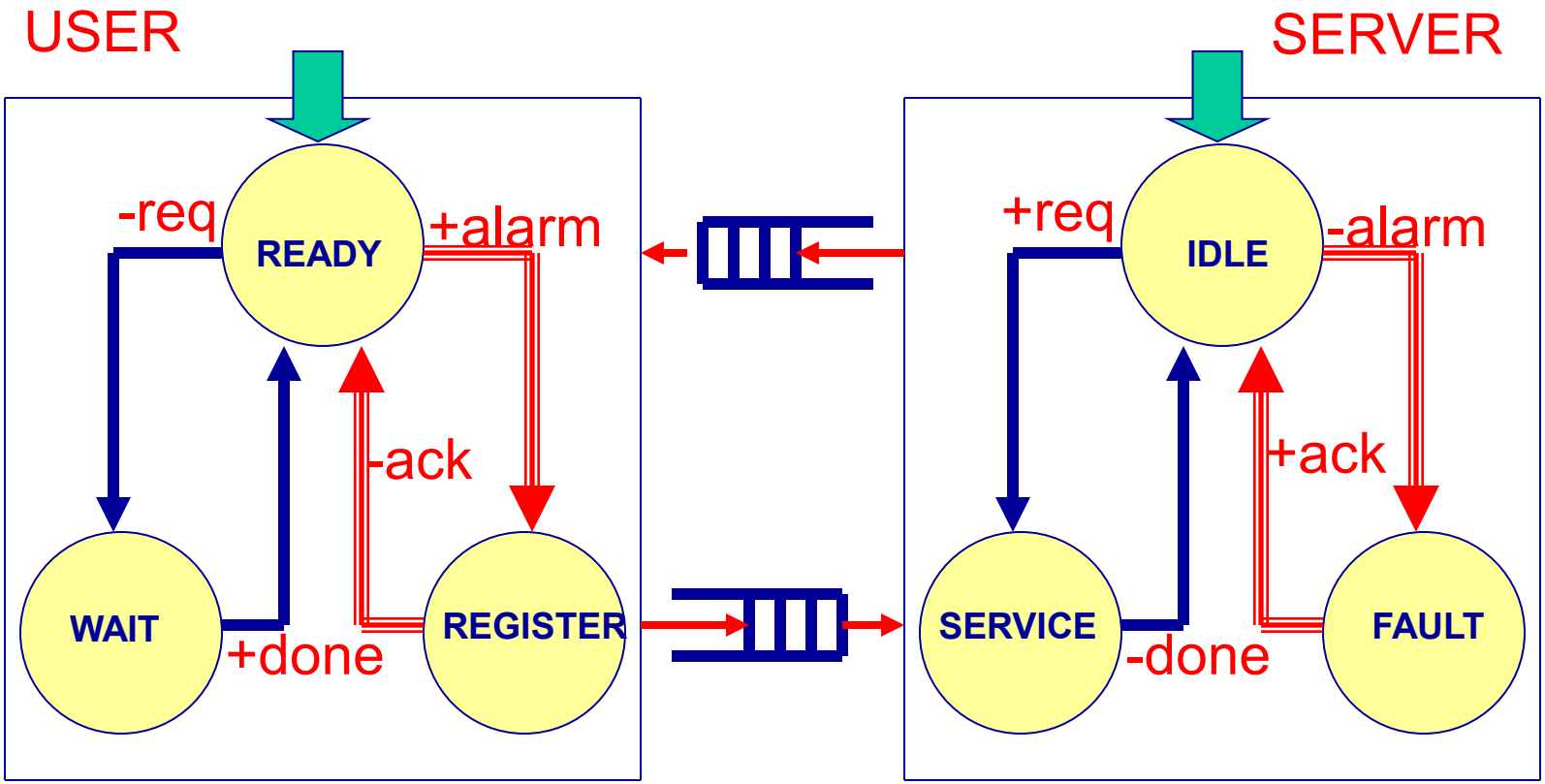


klasyfikacja cech dynamicznych

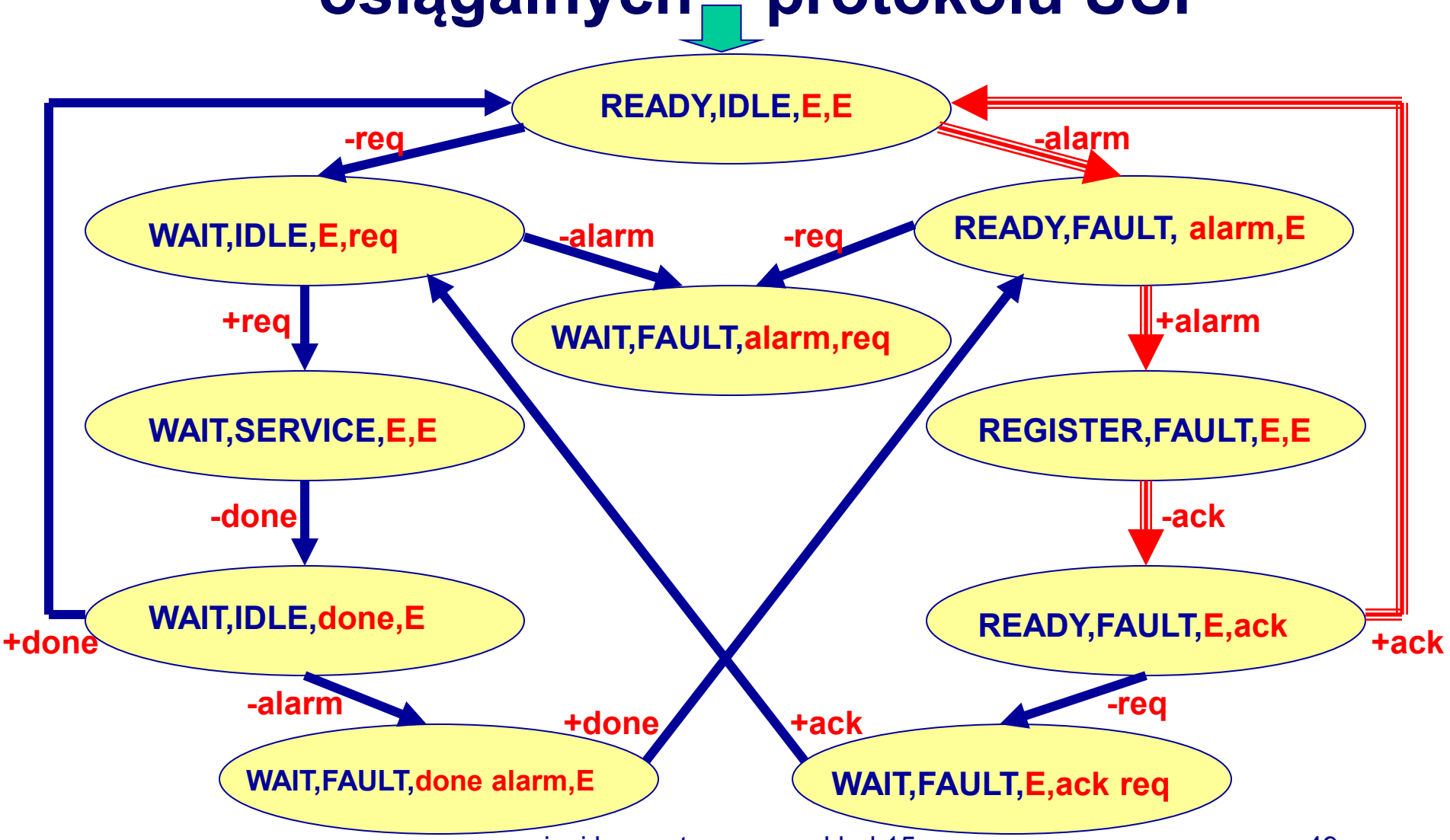
- hierarchiczna, dwustopniowa



blokada dynamiczna w protokole User Server Protocol (USP)



blokada dynamiczna w grafie stanów osiągalnych protokołu USP



blokada dynamiczna - podsumowanie

- wniosek: blokada dynamiczna występuje, gdy cykle w stacjach protokołu tworzą jeden lub więcej cykli w grafie osiągalności, zwanych cyklami blokady dynamicznej [Gouda85]